A goal-oriented fuzzy approach for big data analytics adoption in manufacturing systems

Abstract

Context: The rapid prevalence and potential impact of big data analytics platforms have sparked an interest amongst different practitioners and academia. Manufacturing organisations are particularly well suited to benefit from big data analytics platforms in their entire product lifecycle management for intelligent information processing, performing manufacturing activities, and creating value chains. This requires re-architecting their manufacturing legacy information systems to enable integration with contemporary big data analytics platforms. A systematic re-architecting approach is required incorporating careful and thorough evaluation of goals for big data adoption. In addition, ameliorating uncertainty of the impact the new big data architecture on system quality goals is needed to avoid cost blowout in implementation and testing.

Objective: We propose an approach to reason about goals, obstacles, and to select suitable big data solution architecture that satisfy quality goal preferences and constraints of stakeholders at the presence of the decision outcome uncertainty. The approach will highlight situations that may impede the goals. They will be assessed and resolved to generate complete requirements of architectural solution.

Method: The approach employs *goal-oriented modelling* to identify obstacles causing quality goal failure and their corresponding resolution tactics. It combines fuzzy logic to explore uncertainties in big data solution architecture and to find an optimal set of architectural decisions for the big data enablement process of manufacturing systems.

Result: The proposed approach brings two innovations to the state of the art of big data analytics platform adoption in manufacturing systems: (i) A systematic goal-oriented modelling for exploring goals and obstacles in integrating manufacturing systems with big data analytics platforms at the requirement level and (ii) A systematic analysis of the architectural decisions under uncertainty incorporating stakeholders' preferences. The efficacy of our approach is illustrated with a scenario of reengineering a hyper-connected manufacturing collaboration system to big data architecture.

Keywords: big data, big data analytics platforms, manufacturing systems, goal-oriented modeling, fuzzy approach

1 Introduction

Product lifecycle management is a data intensive process comprising market analysis, product design, development, manufacturing, distribution, maintenance and post-sale, and recycling (Stark 2015). The process involves a variety of voluminous data, e.g. customers' comments on social media, product functions and detailed design specifications, product configuration, and failure incidences reported by installed sensors to monitor parameters of environment and products, etc. Manufacturing organizations view such data as a valuable business asset to achieve good performance and to reduce cost in the product lifecycle. Manufacturing organizations also regularly seek to increase their productivity using new advanced information technologies that place further demand on their data processing storage requirements; such as Internet of Thing (IoT) and radio-frequency identification (RFID) tags in their daily production. For example, Toyota automotive company equip cars with smart sensors and continuously collecting data about its locks, location, ignitions, and tyres which can be later used by the manufacturer assembly. Continuous product innovations lead to further product data generation coupled with great diversity of types, sources, meaning, and format.

Given its increasing volume and variety, manufacturing data is increasingly difficult to process using common manufacturing data platforms be they computer aided design (CAD), supply chain management (SCM) manufacturing execution system (MES), or enterprise resource planning (ERP). Indeed, the high volume, velocity, variety, veracity, and value adding data requirement all point to the need to complement manufacturing systems with *big data* platforms (McAfee, Brynjolfsson and Davenport 2012). New platforms such as Apache Hadoop, Google's Dremel, or S4 are promising ways forward to address the abovementioned complexity of processing (Lycett 2013). They provide a support for capturing, processing, and visualizing large volume of data sets that organizational systems may have collected over years.

Taming big data has the potential added benefit to analyze real-time data across different phases of product lifecycle management, from receipt of a customer's order phase to customer satisfaction phase (Protiviti 2017), (Waller and Fawcett 2013), (Bi and Cochran 2014), (Dubey, Gunasekaran, Childe et al. 2016). The value of big data in the various phases of product lifecycle management has been recently discussed. Phases from identifying promising customers, collecting variable data about the quality of raw material, selecting a detailed design, procuring, selecting suppliers and outsourcing policies, and product warehousing, maintenance, recycling, and to identifying labor errors have been discussed (Li, Tao, Cheng et al. 2015).

Compared to others fields such as electronic commerce, financial trading, health care, and telecommunication, the manufacturing field seems to be slow in pace in adoption big data analytics platforms in their business processes (Li, Tao, Cheng et al. 2015). This could be attributed to the high capital costs associated with manufacturing systems (Camarinha-Matos, Afsarmanesh, Galeano et al. 2009). Nevertheless, manufacturing organizations recognize the value of big data analytics and the fact that their adoption failure poses a risk to their operating and financial performance (Protiviti 2017). The systems automate, process, and integrate data flows between one or more above phases and typically composed of a number of software systems, machines, transportation devices, and so on. Gartner reports that 60 percent of big data projects fail to get piloting and production due to reasons such a lack of adequate IT skill set, inability to understand stakeholders requirements in utilising big data analytics platforms, and disparate legacy systems (Gartner 2015), (Wegener 2013). Thence, reluctance of manufacturing organisations in moving to these platforms is unsurprising. Some are also still figuring out what kind of data is worthy for advanced data analytics and which stage of product lifecycle management is suitable to utilize big data analytics platforms (Govindarajan, Ferrer, Xu et al. 2016), (Jha, Jha, O'Brien et al. 2014), (Bi and Cochran 2014).

It has been a long-standing acknowledgement that a poor system upgrade with a new technology can have far reaching consequences in later stages that are costly to rectify. This continues to be permeating theme in adoptions of big data analytics platforms in manufacturing systems. Particularly, these may involve many competing goals, e.g. security, performance, reliability, scalability, maintainability, and the development cost. There are also unforeseen risks. As articulated by Protiviti: *"Manufacturers should have clear and easily definable goals. As a part of that planning process, companies need to determine whether the systems they have in place will achieve the desired results and/or what enhancements might be required"* (Protiviti 2017). Manufacturing organisations also tend to have their own goals and preferred competitive dimension with respect to taking advantages of big data analytics platforms to augment their systems (Wang, Xu, Fujita et al. 2016). A system architect, who is responsible to the design high-level big data enabled solution architecture for existing systems, should meticulously specify goals of multiple actors, analyze potential risks, and make a right balance among operationalisation of the goals in adopting these technologies (Lee, Kao and Yang 2014) (Wang, Xu, Fujita et al. 2016). For example, using a poor big data visualization technology may negatively affect performance, scalability, and real-time data processing coming from sensors. The choice of a data mining algorithm to process sensor data across the product line may also impact the real-time performance of control systems. An early stage analysis of big data adoption goals gives an opportunity in exploring countermeasures to tackle probable risks in advance rather than drowning in narrow aspects of these technologies. Indeed, failure to do so in early stages can negatively affect the overall manufacturing system performance.

Uncertainties about the impact of decisions on big data adoption goals are unavoidable, as in any other adoption endeavor. That is, a lack of complete knowledge about the actual consequences of decision alternatives is a fact. For instance, the raw feedback data generated by online customers about produced cars that are processed by big data analytics platforms may produce some uncertainties in terms of the interpretation of data. The choice of a big data visualization technique may have an uncertain impact on the reliability of other generated diagnostic reports about a product due to inherent uncertainties of data sources. The architect is still expected to make right choices in such uncertain circumstances. The quest for a risk-aware early stage analysis of big data adoption in making critical and uncertain decisions remains a top priority as highlighted in (Pal, Meher and Skowron 2015) and (Chen and Zhang 2014).

This paper provides a systematic approach aiding system architects for goal-obstacle analysis of big data solution architectures and selecting architectural decisions using imperfect information. It provides a step-by-step goal-obstacle analysis process to address uncertain risks. It then produces a complete set of requirements, ranks candidate architectures based on the fuzzy logic, and ultimately to find an optimum architecture. The contributions of the paper are thus two folds: (i) It provides a goal-oriented approach for reasoning about architectural requirements at the early stage of adoption (ii) It deals with uncertainty about the impact of integrating big data analytics platform on manufacturing system which can be unforeseen at requirement time. The rest of this article is organised as follows: Section 2 provides the background of this study including a motivating scenario and fundamental concepts used in the proposed approach. Section 3 details the proposed approach. Section 4 illustrates the approach in a scenario of re-architecting a legacy car manufacturing system to incorporate big data analytics platforms. Section 5 outlines other related studies. Finally, Section 6 summarises the article with a discussion of limitations and future research directions.

2 Background

For illustrating the goal modelling and fuzzy set theory and other concepts in this article, we adopt and extended a reengineering scenario of a hyper-connected manufacturing collaboration system (HMCS) (Lin, Harding and Chen 2016). HMCS addresses challenges of extracting, processing, and analysing data from multiple distributed web sources. In HMCS, interoperability and processing of heterogeneous digital data is supported via the semantic web. Multiple big data analytics platforms found a viable enabler to provide a great support for data extensive processing service for a different variety and volume of both structured and unstructured data. This scenario is first presented.

2.1 A motivating scenario of moving manufacturing systems to big data platforms

In the scenario used in this paper, HMCS provides a platform to enable several partners of Toyota car manufacturing to collaborate and to share knowledge about car products and parts across product line. At the core of the HMCS architecture, a data Extract-Load-Transform (ELT) process integrates data streams from multiple manufacturing parties. The ETL process integrates data from different types of databases, including those storing data coming from sensors in the product line or from buyers. The process has a middleware layer that includes rules and logic for mapping data between different formats. An additional data stream comes the online Toyota buyer conversations that appear in social media, such as Twitter, posts, Internet server logs, and blogs. This stream provides feedback on recent purchase experiences, warranty claims, repair orders, service reports and others. This data stream provides early customer feedback to uncover actionable trends and to generate appropriate early-warning signals to the manufacturing process. This stream is increasingly voluminous. It generates millions of items per day.

The heterogeneity of data sources and the large volume of data strain the ETL. It often becomes a bottleneck and incapable of identifying all patterns and generating the statistical reports. To resolve this, the IT department of Toyota aims to deploy multiple big data analytics platforms. The aim is to enable extraction and management of both sources of data, i.e. internal manufacture parties and the unstructured data in the online Toyota buyer conversations. A system architect is appointed specifically to design a solution to upgrade the ETL and integrate various new services offered by the data analytics platforms. An immediate concern of the architect is a cost benefit analysis of the adoption of the platforms evaluating the risks and mapping the way forward. The following questions are pertinent to the architect: What are ETL system quality goals? How these may affected (positively or negatively) if big data analytics platforms are utilized? By adopting such platforms, will higher ETL performance be attainable in all circumstances? What obstacles are likely to occur during and after re-engineering HMCS to big data analytics platforms and what are their severity? What countermeasures can be added in advanced to negate such obstacles and do they have any side effects? Answering these questions is a challenging task as it involves reasoning with a long chains of 'what-if' scenarios and their uncertain impacts on goals given by stakeholders in HMCS.

Additionally, these impacts are may be hard to quantify or even imprecise. Human judgments are often too vague for using exact numerical values, let alone in an early stage of solution architecture design.

Towards migrating to a big data solution architecture, we offer an approach that explicitly relates manufacturing system high-level quality goals to potential obstacles, highlights alternatives in addressing them and assesses their impact on stakeholders' goals, calculates uncertainty of the various impacts, and finally shortlists candidate architectures based on the goals.

2.2 Goal-oriented requirement modeling

Goal-oriented reasoning approaches such as KAOS, are means for elicitation, elaboration, and analysis of system requirements (Yu and Mylopoulos 1994). They avoid prematurely delving into technical and implementation aspects of big data analytics platforms. A system architect can instead seek a wide range of resolutions and can negotiate them with the stakeholders. We choose the commonly used KAOS framework which includes two components: (i) a modeling language defining concepts such as goal, obstacles, agents, operations, and domain objects and (ii) a method including a series of steps to elaborate and analyze different goals (Van Lamsweerde 2009). In this section, we present the key KAOS concepts that are used in this paper. Further details on KAOS can be found in Lamsweerde's book (Van Lamsweerde 2009).

A *goal* is a prescriptive statement of intention that a system should satisfy through the cooperation of *agents*. A *goal* has a name and a specification expressed using natural or formal languages. The specification defines what the goal means and its satisfaction conditions. Goals may range from high-level business objectives to fine-grained technical ones. In our context, we mean common system quality goals such as performance, security, and maintainability. In a goal model, goals can be refined into fine granular ones through decomposition. The method part of the KAOS framework provides a process to create a system requirement goal through a hierarchical refinement process. All goals are continuously refined into sub-goals until all sub-goals can be assigned to a single stakeholder (a user or a system component).

A goals can be viewed optimistic vantage point that overlooks unexpected condition of a real environment that may cause their failures (van Lamsweerde and Letier 2000; Letier 2001). However, taking a more realistic, these conditions are construed as *obstacles*. Obstacles are duals of goals in the sense that as goals represent desired conditions, obstacles represent undesirable conditions (Letier 2001) and should be systematically identified. They need to be assessed and tackled via defining resolution tactics at an early stage of a system development to identify any needs to modify the goals (Letier 2001). As such, in KAOS, resolving obstacles includes generating and selecting alternatives to resolve obstacles. Selection of a subset of decision alternatives satisfying goals faces a multi-criteria decision making problem (MCDM) with the possibility of different priority of goals in view of stakeholders needs. Any uncertainty in selecting options may also occur in terms of a range of impacts that an alternative decision may have on goals. It is often the case that stakeholders may express such impacts qualitatively or using imprecise measures because their judgements are unavoidably vague and indescribable with exact numerical values at the early stage of the project. For example, the impact of choosing a data mining algorithm for processing data coming from sensors might be expressed in linguistic terms or a range of values rather than a crisp and single number. Subsequently, each decision alternative may fall within a range. Comparing two decision alternatives with overlapping in their impact on goals is not easy. There is often a need to consider trade-offs amongst various alternatives. Whilst MCDM frameworks can help in comparing, prioritizing, and selecting the most suitable resolution tactics, they do not reflect uncertainty in human thinking style. Fuzzy logic can better handle uncertainty. For instance, expressions such high performance or low cost become usable. We make use of synergies between the KAOS approach and fuzzy logic to cope with various sources of uncertainties in integrating manufacturing systems with big data analytics platforms.

2.3 Fuzzy set theory

Fuzzy set theory analogizes human judgment in its use of proximate information and uncertainty in decision making (Zadeh, Fu and Tanaka 2014). Whilst classic sets define crisp values, fuzzy set theory shows groups of data with boundaries that are not crisp. This provides a better capability to resolve real-world problems, which unavoidably involve imprecise and noisy parameters. Accordingly, linguistic expression of variables is the central aspect of fuzzy logic where general terms such a *very large*, *large*, *medium*, *small*, *too small* are used to represent a range of numerical values.

Instead of showing the anticipated impact of an architectural alternative on system quality goals as a discrete point, we represent such impact as a range of values. This is more aligned with human judgment in conceptualization and representation of uncertainty. The impact range of each alternative can be based on statistical data from similar past decision making situations or expert opinions.

The ability to quantitatively analyse alternatives in a big data solution architecture manufacturing systems can be achieved via representing uncertain parameters as *fuzzy numbers* which belong to *fuzzy sets*. The fuzzy set, originally proposed by Zadeh, is defined as follows (Zadeh, Fu and Tanaka 2014): In a universe of discourse U, a fuzzy subset A is characterized by a membership function F, where F: U – and membership function is associated with each member of x of U is a number of F in the interval [0,1] denoting the membership of x in A. We define the impact of decision alternatives as 1 to the anticipated value, and possibility of 0 to the optimistic and pessimistic, respectively. We use linguistic variables to represent anticipated, optimistic, and pessimistic impact of alternatives on system quality goals, respectively. The impact linearly decreases from very low to very high. This range of impact is represented using a triangular fuzzy value (Pedrycz 1994). For example, the choice of a particular data mining algorithm for processing sensor data may have *Very High* positive impact on the overall system performance. Such values may be available from similar architecture designs in other systems, system architect's experience, or expert judgment.

3 The approach

Our approach, shown Figure 1, has two steps: High-level goals for big data analytics platforms, operational decision alternatives, and potential obstacles are first identified. The obstacles are then assessed. Any severe obstacles are resolved through prescribing resolution tactics. This step engages the stakeholder and leads to iterative goal model refinement using techniques similar to those described in (Lim and Finkelstein 2012). Alternatives decisions are likely adopted to operationalize goals and to resolve obstacles. Their impact on big data adoption goals are then analysed to find optimize the goal achievement. The overall output of the framework is a set of solution architectures ranked on the basis of satisfying quality goals. The final selected architecture can later be incorporated to reengineer manufacturing systems using big data analytics platforms. To illustrate the inner working of these steps, the scenario presented in Section 2.1 is used.

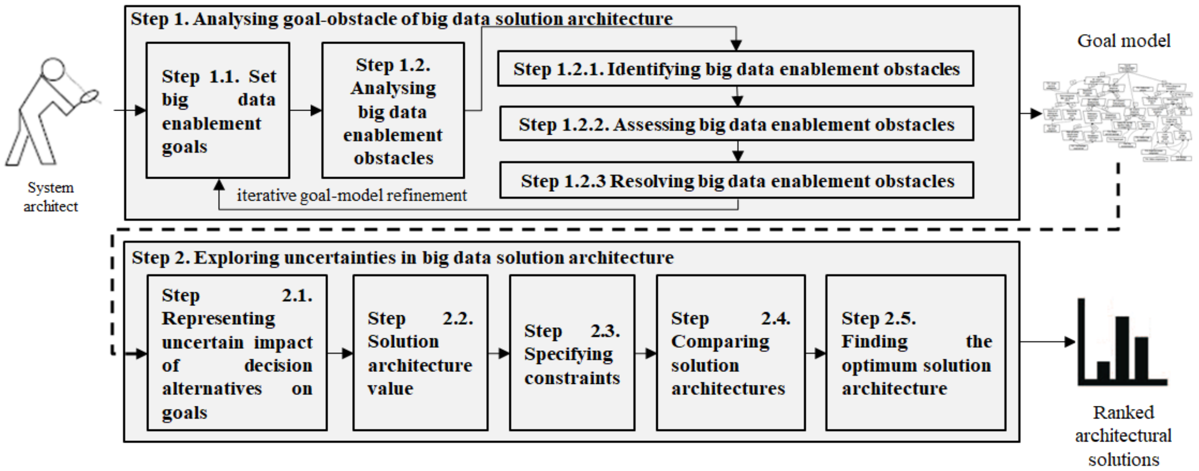
Step 1. Analysing goal-obstacle of big data solution architecture
Step 1.1. Set big data enablement goals
Step 1.2. Analysing big data enablement obstacles
Step 1.2.1. Identifying big data enablement obstacles
Step 1.2.2. Assessing big data enablement obstacles
Step 1.2.3 Resolving big data enablement obstacles
iterative goal-model refinement
System architect
Goal model
Step 2. Exploring uncertainties in big data solution architecture
Step 2.1. Representing uncertain impact of decision alternatives on goals
Step 2.2. Solution architecture value
Step 2.3. Specifying constraints
Step 2.4. Comparing solution architectures
Step 2.5. Finding the optimum solution architecture
Ranked architectural solutions

Figure 1. Proposed approach

3.1 Step 1. Analyzing goal-obstacle of big data solution architecture

The first step is based on KAOS modelling framework and deploys the notation shown in Table 1. Step 1 includes these two sub-steps:

(i) *Step 1.1. Specifying big data adoption goals* specifying and visualizing high-level quality goals targeting by adopting big data analytics platforms and, possibly, decision alternatives to operationalize goals,

(ii) *Step 1.2. Analysing big data adopting obstacles* for identifying obstacles causing goal failure, assessing their likelihood and criticality of consequence, and defining resolution tactics, another set of decision alternatives, to modify existing goals, generate new goals to prevent, to reduce, or to mitigate the obstacles.

Table 1. notations used for goal modelling

Modelling element	Definition	Graphical notation
Root (overall goal)	The overall goal of adopting big data analytics platforms.	⬢
Goal	A quality goal that is expected to be satisfied by utilizing big data analytics platforms.	🟦
Obstacle	A technical or a none-technical exceptional situation/condition preventing the goal satisfaction.	🟥
Architectural decision	A generic architectural solution either to operationalize a goal or to tackle an obstacle.	🟩
Contribution	Positive contribution of adopting big data analytics platform.	—+→
Decomposition	A mechanism to refine goal/obstacle to set of sub goals/obstacles.	⸺/→
Architectural decision alternative	An alternative either to operationalize a goal or to tackle an obstacle.	-·-+-·→

The following subsections provide technical details of Step 1.

Step 1.1 Set big data enablement goals. Goals are introduced as means to identify stakeholders' needs at the early stage of requirement analysis. In our scenario, the following seven goals are set for the integrating the system with big data platforms (Figure 2): *Achieve [Improved performance]*, *Achieve [Improved availability]*, *Achieve [Maintained interoperability with other big data platforms]*, *Achieve [Improved data visualization]*, *Achieve [Maintained data security on big data platforms]*, and *Achieve [Increased unstructured data storage capacity]*. The goal *Achieve [Improved performance]*, is further decomposed into sub-goals *Achieve [Processed social media weekly under expected time]* and *Achieve [Processed sensor data under expected time]*. That is, the satisfaction of *Achieve [Improved performance]* depends on the satisfaction of both these sub-goals.

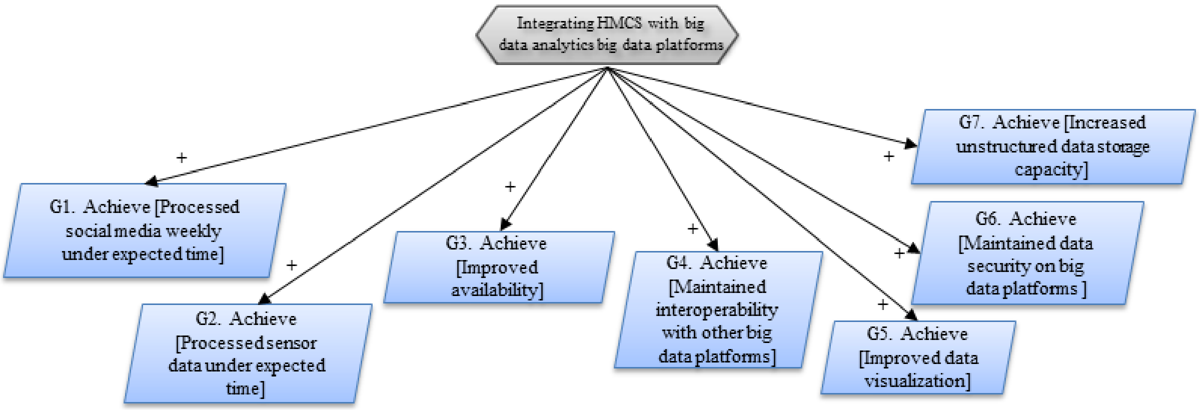
Integrating HMCS with big data analytics big data platforms
+
G1. Achieve [Processed social media weekly under expected time]
+
G2. Achieve [Processed sensor data under expected time]
+
G3. Achieve [Improved availability]
+
G4. Achieve [Maintained interoperability with other big data platforms]
+
G5. Achieve [Improved data visualization]
+
G6. Achieve [Maintained data security on big data platforms]
+
G7. Achieve [Increased unstructured data storage capacity]

Figure 2. six goals for integrating HMCS with big data analytics platform

In our scenario, HMCS databases rapidly grow in size and reach several terabits of data collected by sensors in product line. An unlimited data storage capacity is required. The specification of goal *Achieve [Increased unstructured data storage capacity]* is documented as follows:

Goal Achieve [Increased unstructured data storage capacity]

Category scalability goal

Definition Batches of data records from manufacture product line provided by installed sensors should be stored continuously. These records consist of data monitored by robots about assembling Toyota parts in the production line.

Quality Variable storageSize: Batch → Size

Definition The required capacity in storing records of data collected by sensors in a working day.

Sample Space The set of daily cars is assembled and delivered to the end of the product line.

Objective Functions At least one gigabyte of records (e.g. images, events, logs, and errors) are generated at the end of a working day. The database should be able to store this volume.

For the goal *Achieve [Processed social media weekly]*, the following definition is documented:

Goal Achieve [Processed social media weekly under expected time]

Category performance goal

Definition Relevant data to buyers experience should be collected from from Twitter, posts, Internet server logs, and blogs and then be processed every week and be available on Monday at 12am. Processing consists of generating feedback, complaints, fix requests, ordering parts, comparisons by other cars that have been made by buyers.

Quality Variable ProcessedTime: Batch → Time

Definition The required capacity to storage sensor data at the end of the day.

Sample Space The set of daily cars that are assembled on the product line and delivered.

Objective Functions The processing all the collected data and generating reports should be started from 12 am on Saturday and finished at 12 on Sunday.

Goals can be operationalized through different architectural alternatives, i.e. tactics, techniques, and technologies. As shown in Figure 3, to operationalize the goal *Achieve [Increased unstructured data storage capacity]*, the system architect may consider five mainstream big data stores namely *MongoDB*, *Memchased*, *HBase*, *Cloudant*, and *BigTable*. Likewise, to realize the goal *Achieve [Processed social media weekly under expected time]*, both operational technologies *scheduler* and *social media data processing* can be implemented and operationalized where each has different alternatives to be employed (Figure 3).

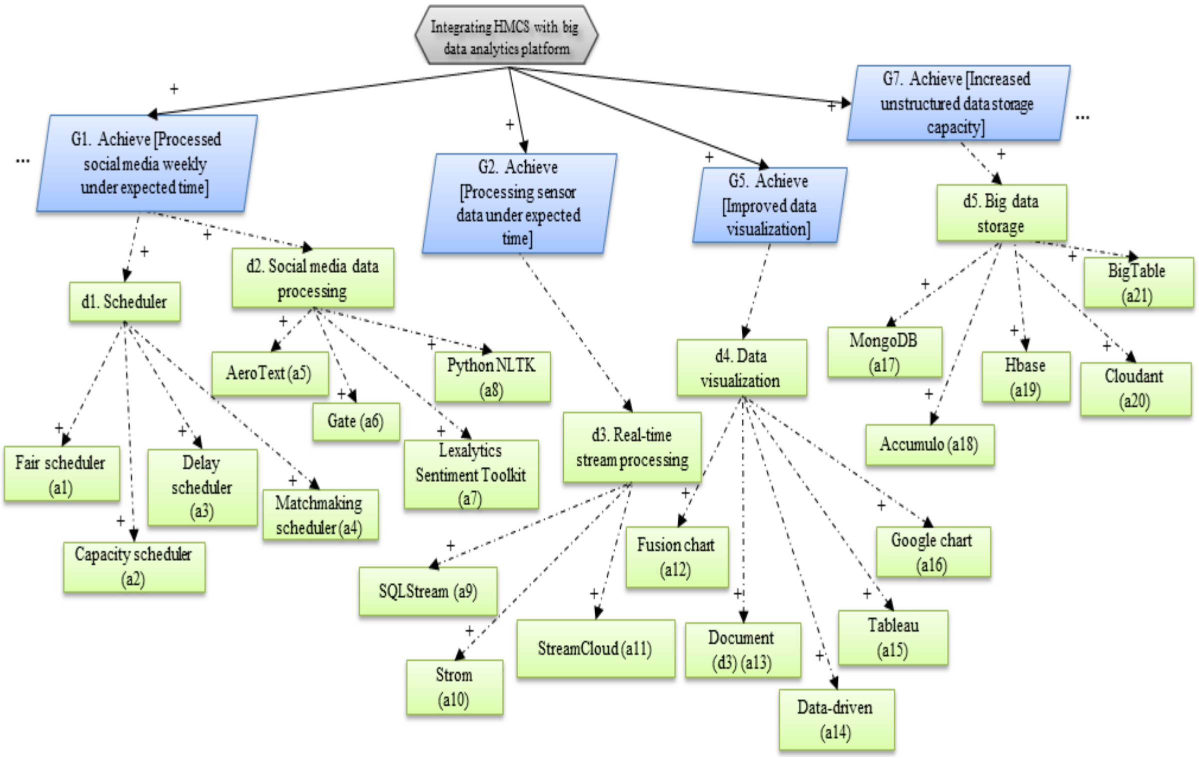
Integrating HMCS with big data analytics platform
G1. Achieve [Processed social media weekly under expected time]
...
G2. Achieve [Processing sensor data under expected time]
G5. Achieve [Improved data visualization]
G7. Achieve [Increased unstructured data storage capacity]
...
d1. Scheduler
d2. Social media data processing
d3. Real-time stream processing
d4. Data visualization
d5. Big data storage
Fair scheduler (a1)
Capacity scheduler (a2)
Delay scheduler (a3)
Matchmaking scheduler (a4)
AeroText (a5)
Gate (a6)
Lexalytics Sentiment Toolkit (a7)
Python NLTK (a8)
SQLStream (a9)
Strom (a10)
StreamCloud (a11)
Fusion chart (a12)
Document (d3) (a13)
Data-driven (a14)
Tableau (a15)
Google chart (a16)
MongoDB (a17)
Accumulo (a18)
Hbase (a19)
Cloudant (a20)
BigTable (a21)
+

Figure 3. partial model for operationalization of the goals through alternative architectural decisions

Step 1.2. Big data enablement obstacle analysis. Normally, goals that are represented neglect unexpected situations that can cause their failures in operational environment (van Lamsweerde and Letier 2000; Letier 2001). As mentioned earlier, these situations are referred to as obstacles. They should be systematically identified, assessed, and mitigated against. This may lead to goal model elaboration. If goals are not threatened by any obstacles, the system architect can proceed to Step 2 (detailed in Section 3.2). Otherwise, defining an iterative identify-assess-resolve cycle for the obstacle analysis is required as follows:

(i) Identifying obstacles that may impede the goals' satisfaction (Step 1.2.1);
(ii) Assessing obstacles' risks in terms of their likelihood and criticality (Step 1.2.2); and
(iii) Resolving obstacles via modifying existing goals or generating new ones in order to prevent, reduce, or mitigate obstacle occurrence (Step 1.2.3).

Step 1.2.1. Identifying big data enablement obstacles. Obstacles originate from intrinsic characteristics of big data analytics platforms or their operations. The system architect uses domain information to iteratively refine the goal model identifying obstacles and any sub-obstacles (Letier 2001). In our scenario, the candidate big data store technologies for operationalization of goal *Achieve [Increased unstructured data storage capacity]* may obstruct goal *Achieve [Maintained interoperability]* (Figure 4). The reason is that existing HMCS's databases are relational. These are not compatible with no-SQL based and schema-free data storages such as *MongoDB*, *Memchased*, *HBase*, *Cloudant*, and *BigTable*. Converting thousands of line of complex T-SQL and analytical codes in HMCS to this type of big data storages for intensive processing is not a simple task. In other words, the alternative technologies raise the *obstacle Incompatibility of legacy and cloud service (O48)*. This obstacle is further decomposed into sub-obstacles *Incompatibility of legacy data storage and cloud (O49)* and *Incompatible APIs (O44)*. The obstacle *Incompatibility of legacy data storage and cloud (O49)*, is itself a combination of the obstacles *Incompatible datatypes (O21)* and *Incompatible data operations (O50)*. Big data analytics platforms leverage cloud computing servers which are often vulnerable to issues such as bandwidth capacity bottleneck, performance variability or scaling latency, and security (Agrawal, Das and El Abbadi 2011). The partial goal model in Figure 5 represents probable obstacles against the goals that are identified by the system architect.

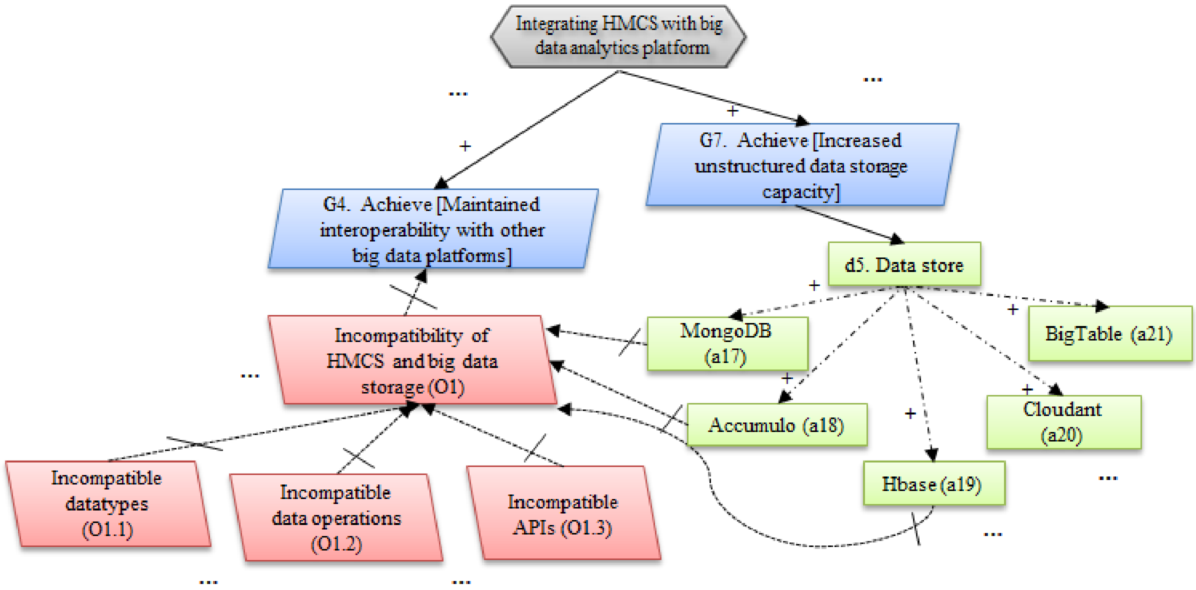
Integrating HMCS with big data analytics platform
...
...
+
+
G7. Achieve [Increased unstructured data storage capacity]
G4. Achieve [Maintained interoperability with other big data platforms]
d5. Data store
+
+
MongoDB (a17)
BigTable (a21)
Incompatibility of HMCS and big data storage (O1)
...
+
+
Accumulo (a18)
Cloudant (a20)
+
Hbase (a19)
...
Incompatible datatypes (O1.1)
Incompatible data operations (O1.2)
Incompatible APIs (O1.3)
...
...
...

Figure 4. obstacles to the goal *Achieve [Maintained interoperability with other big data platforms]* if big data stores are employed

Step 1.2.2. Assessing big data enablement obstacles. The identified obstacles from Step 1.2.1 are assessed to get an understanding of a new set of architectural alternatives for integrating HMCS with big data analytics platforms. Criticality of the obstacles is judged based on their impact on the goals. Qualitative and quantitative techniques can be employed to perform this step. Our approach employs a common qualitative technique, Risk Analysis Matrix (Franklin 1996). This specifies the likelihood of an obstacle using a qualitative scale ranging from Almost Certain, Likely, Possible, Unlikely, and

Rare. It also specifies the obstacle consequence as Insignificant, Minor, Moderate, Major, and Catastrophic. The risk of an obstacle is defined as the product of its occurrence and severity, i.e. Risk = Likelihood × Consequences. Estimating this risk relies on the availability of domain information sources such as statistics from manufacturing systems, existing accounts on big data analytics platforms, or system architect's judgement. The system architect may conduct a voting technique involving all stakeholders to assess probability occurrence and severity of obstacles. A risk matrix highlights the risk zone as shown in Table 2. For instance, the risk of an obstacle might be considered as moderate (M), however, it is still tolerable. Whilst an obstacle with H (High) and E (Extreme) necessitates a proper architectural decision. The values represented in Table 2 are exemplar values specified regarding about the HMCS domain information.

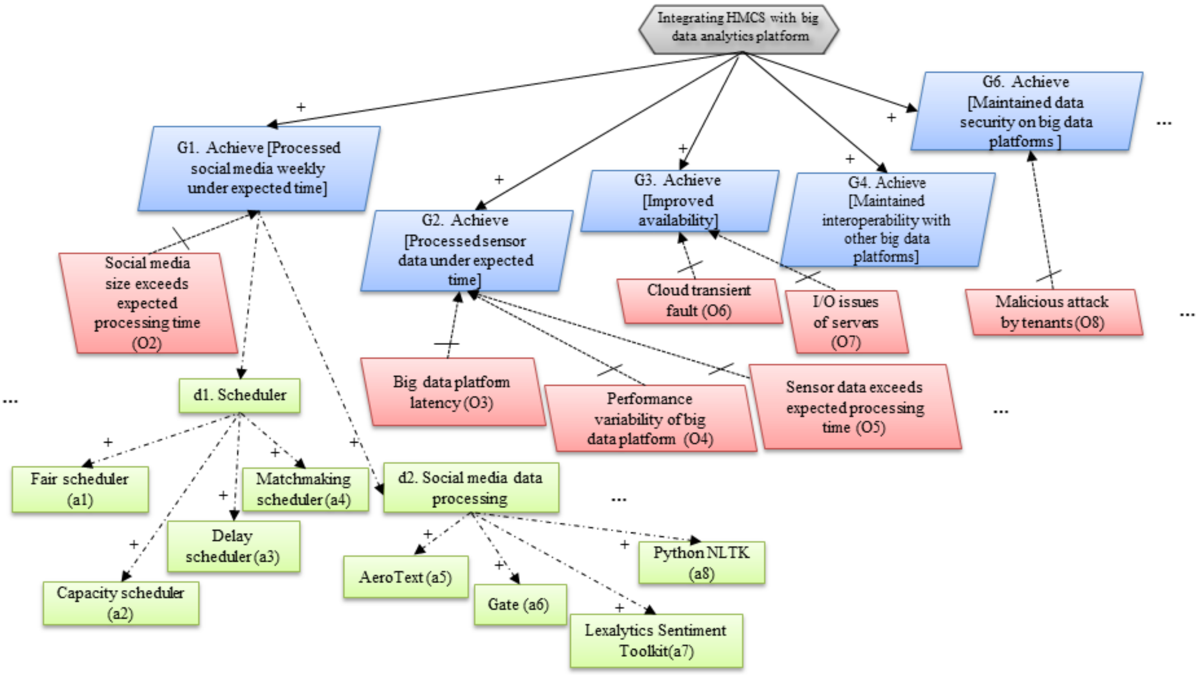
Integrating HMCS with big data analytics platform
G1. Achieve [Processed social media weekly under expected time]
G2. Achieve [Processed sensor data under expected time]
G3. Achieve [Improved availability]
G4. Achieve [Maintained interoperability with other big data platforms]
G6. Achieve [Maintained data security on big data platforms]
Social media size exceeds expected processing time (O2)
Big data platform latency (O3)
Performance variability of big data platform (O4)
Sensor data exceeds expected processing time (O5)
Cloud transient fault (O6)
I/O issues of servers (O7)
Malicious attack by tenants (O8)
d1. Scheduler
d2. Social media data processing
Fair scheduler (a1)
Capacity scheduler (a2)
Delay scheduler (a3)
Matchmaking scheduler (a4)
AeroText (a5)
Gate (a6)
Lexalytics Sentiment Toolkit(a7)
Python NLTK (a8)
+
...

Figure 5. identified probable obstacles to goals

Table 2. risk matrix for obstacles

	Consequence severity				
Likelihood	Insignificant	Minor	Moderate	Major	Catastrophic
Almost Certain	H	H	E	E	V
Likely	M	H	H	E	V
Possible	L	M	H	E	E
Unlikely	L	L	M	H	E
Rare	L	L	M	H	H

V: Very extreme risk, E: Extreme risk; H: High risk; M: Moderate risk; L: Low risk

Step 1.2.3 Resolving big data enablement obstacles. Obstacles deemed with severe risk are resolved. This requires generating architectural alternatives and selecting suitable alternatives amongst them. We employ KAOS's obstacle resolution tactics as mentioned earlier. KAOS defines eight generic and platform independent resolution tactics: *goal substitution*, *agent substitution*, *obstacle prevention*, *goal weakening*, *obstacle reduction*, *goal restoration*, *obstacle mitigation*, and *do-nothing* (van Lamsweerde and Letier 2000; Letier and Van Lamsweerde 2004). These are operators on a goal model to refine it to new or existing modified goals, assumptions, and responsibility assignments. They are defined as follows.

(i) Substitute goal defines a new alternative goal which is still contributable by big data analytics platforms in a way that the obstacle is no longer present. Consider the performance goal *Achieve*

[Processed social media weekly] obstructed by the obstacle *Social media size exceeds processing speed time(O)*. An instance of this tactic is to collect and processes data daily, instead of weekly.

(ii) Substitute big data analytics platform removes the occurrence of an obstacle by replacing the responsibility for an obstructed goal to a new platform. For example, the obstacle *Sensor data processing exceeds expected time (O)* can be removed via transferring the assigned goal from an overloaded server to another server with lower workload.

(iii) Prevent obstacle introduces new assertions to a goal model preventing the obstacle occurrence via applying some factors or doing things in particular way. For instance, consider the security obstacle *Malicious attack by tenants (O3)* obstructing the goal *Achieve [Maintained security of sensor data]* (Figure 6). An application of this tactic is to encrypt the batch data collected from sensors prior storing them on big data storages. As such, bath data cannot be read or processed by malicious tenants that are in performing on the same cloud servers. The system architect considers three architecture decision alternatives *Obfuscate* data, *Redact* data, and *Mask* data. Furthermore, to prevent the occurrence of obstacles *Incompatible datatypes (O21)* and *Incompatible data operations (O50)*, architecture decisions *Adapt data* and *Develop adaptor* are considered. It should be noted employing architecture decision alternatives may cause another set of obstacles against goals. For instance, in the one hand the system architect considers *Adapt data* and *Develop adaptor*. One the other hand, these alternatives may negatively influence the goal *Achieve [Reduced processing sensor data]*. These dependencies are modelled in Step 2 of the approach described in Section 3.2.

(iv) Reduce obstacle introduces agents such as human or servers to behave in certain ways to lessen the occurrence likelihood of an obstacle. Consider the obstacle *Sensor data exceeds expected processing time (O3)* to the goal *Achieve [Reduced processing sensor data]*. An example of applying this tactic is to reduce server workload by prioritizing upcoming data are sent by sensors installed in product line. The data from highly important sensors are collected and processed take precedence over those sensors providing supplementary data or don not need a real-time processing. In addition, to reduce the likelihood occurrence of the obstacle root obstacle *Performance variability of big data platform (O3)*, the architectural decision is *Refine network topology*. Finally, as mentioned earlier, big data analytics platforms may be vulnerable to issues such as server latency as represented by the obstacle *Big data analytic platform latency* in Figure 6. To reduce its effect, the architectural decision *Acquire more resources* (e.g. virtual machines) is chosen.

(v) Weaken goal suggests degrading the goal definition to make it more liberal and relaxed in a way that the obstruction is no longer occurs. This can be done in two ways

-relaxing assumptions of an obstructed goal so that its original form does not needs to be satisfied in all situations. The goal *Achieve [Reduced processing sensor data]* obstructed by *Sensor data size exceeds expected processing time (O3)* is modified to one that the goal is not required to be satisfied in all situations, particularly when server running big data analysis is not in its fully capacity. The goal definition is the same as original one but the difference is the degree of goal degrading.

-relaxing required levels of goal satisfaction condition, meaning that there is no further need to full satisfaction. One example of applying this tactic to goal *Achieve [Reduced processing social media weekly under expected time]* is to soften its definition to maximum acceptable time to be achieved by increasing the time/date, i.e. quality variable *processedTime*.

(vi) Restore goal and mitigate obstacle are two tactics usable when the avoidance of all obstacles is too costly and tolerating or mitigating consequences of obstacles becomes more practical. In the goal restoration tactic, the system architect adds a new goal to the goal model to give back satisfaction condition of the goal ultimately. For the obstacle mitigation, the system architect adds a new goal to attenuate the aftereffects of the obstacle occurrence. The tactic intends achieving a weaker satisfaction of an obstructed goal. In the discussed scenario, the goal *Achieve [Improved availability]* is obstructed by the obstacle *Cloud transient fault* as big data platforms leveraging cloud might be temporarily unavailable due to reasons such as network traffic or server workload. An architecture decision to mitigate this obstacle is to introduce a new goal *Achieve [Retry connection]* in system architecture assuring a weaker version of the goal specifying the delay before next retrying to connect to the server when transient faults occur. Figure 6 shows the goal model after introducing resolution tactics.

(vii) Do nothing accepts the risk of the occurrence of the obstacle.

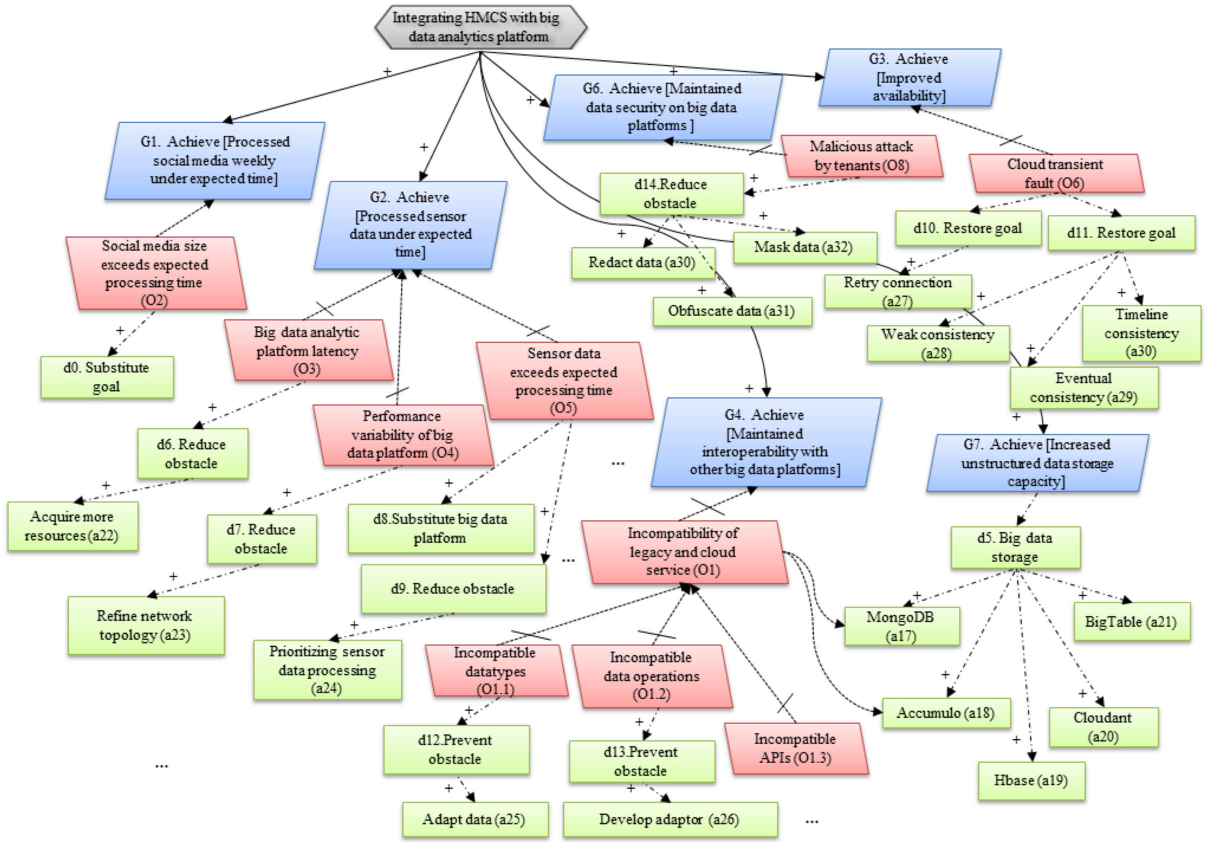
Integrating HMCS with big data analytics platform
G1. Achieve [Processed social media weekly under expected time]
G2. Achieve [Processed sensor data under expected time]
G3. Achieve [Improved availability]
G4. Achieve [Maintained interoperability with other big data platforms]
G6. Achieve [Maintained data security on big data platforms]
G7. Achieve [Increased unstructured data storage capacity]
Social media size exceeds expected processing time (O2)
Big data analytic platform latency (O3)
Sensor data exceeds expected processing time (O5)
Performance variability of big data platform (O4)
Malicious attack by tenants (O8)
Cloud transient fault (O6)
Incompatibility of legacy and cloud service (O1)
Incompatible datatypes (O1.1)
Incompatible data operations (O1.2)
Incompatible APIs (O1.3)
d0. Substitute goal
d5. Big data storage
d6. Reduce obstacle
d7. Reduce obstacle
d8.Substitute big data platform
d9. Reduce obstacle
d10. Restore goal
d11. Restore goal
d12.Prevent obstacle
d13.Prevent obstacle
d14.Reduce obstacle
Acquire more resources (a22)
Refine network topology (a23)
Prioritizing sensor data processing (a24)
Adapt data (a25)
Develop adaptor (a26)
Retry connection (a27)
Weak consistency (a28)
Eventual consistency (a29)
Timeline consistency (a30)
Redact data (a30)
Obfuscate data (a31)
Mask data (a32)
MongoDB (a17)
Accumulo (a18)
Hbase (a19)
Cloudant (a20)
BigTable (a21)
+
...

Figure 6. decision alternatives as result of resolution tactics for handling obstacles

The generated architectural alternatives either operationalize goals or address obstacles. They form a solution space of different architectures to integrate a manufacturing system with big data analytics platforms. Selecting a suitable solution architecture is a challenging task. This is dealt with systematically next in Step 2 of the proposed approach.

3.2 Step 2. Exploring uncertainties in big data solution architecture

This step explores and selects candidate solution architecture under uncertainty. It consists of the five sub-steps and their variables presented in Table 3.

Table 3. symbols used in exploring uncertainty for step 2

Symbols	Definition
g	A goal
G	Set of goals
a	An decision alternative
A	Set of all decision alternatives
D	A set of architectural decisions
d	A decision
x_a	If an decision alternative is selected then 1, otherwise 0
$arch$	An solution architecture including decision alternatives
AS	All possible solution architectures based on architectural decisions and alternatives
$\widetilde{con}_{g,a}$	the contribution (positive or negative) of an alternative *a* on a goal *g*
$\widetilde{S_g}(arch)$	The total value of a solution architecture with respect to all goals
P_g	The weight of the goal *g*
Thd_g	A threshold/constraint to goal *g*
Thd_c	A threshold/constraint for the cost of solution architecture
G_{max}	A goal *g* is aimed to maximized
G_{min}	A goal *g* is aimed to minimized
$\widetilde{cost}_a$	Fuzzy cost of the alternative *a*

$CH^k(\widetilde{S}\,(arch_i))$	Ranking index of solution architecture *i* with total value $\widetilde{S}$

Step 2.1. Representing uncertain impact of decision alternatives on goals. We define all architectural decisions as set *D*. As mentioned in Step 3.1, the choice for operationalization goal *Achieve [Increased unstructured data storage capacity]* through different big data stores is one such decision (Figure 6). Each decision $d \in D$ may have alternatives for operationalization, which is specified using set A_d. For instance, the decision on *big data storage* five alternatives, namely: *MongoDB*, *Accumulo*, *HBase*, *Cloudant*, and *BigTable* (Figure 6). As described in Step 3.1, a decision and its alternatives can be derived using resolution tactics. In our scenario, the architecture decision *prevent obstacle* to handle the obstacle *Malicious attack by tenants (O3)* generates three different alternatives *Mask data*, *Mask data*, and *Obfuscate*. This set of alternatives is defined A = $\cup_{d\in D}\ A_d$. A candidate solution architecture, AS, is an adequate subset of all possible decision alternatives. Thus, AS is represented as follows:

$$\overset{\text{def}}{=} \{arch \subseteq A | (\forall\, d \in D : \exists\, a \in A_d : a \in arch\,) \wedge (\forall\, a \in arch,\ a \in A_d : \nexists\, b \in A_d : b \neq a \wedge b \in arch)\}$$

For a decision alternative $a \in A$ and goal $g \in G$, we define $\widetilde{con_{g,a}}$ specifying the contribution/impact of the alternative *a* on the goal *g*. The symbol ~ indicates that the contribution is a fuzzy number. To represent a fuzzy impact, our approaches uses *Triangular fuzzy numbers* (TFNs) (Pedrycz 1994). TFNs are widely used to represent the approximate value range of linguistic variables. A triangular fuzzy number is represented by A = (a, b, c) where the parameters a, b, and c, respectively, shows the smallest possible value, the most promising value, and the largest possible value describing a fuzzy event. TFN linear membership function μA is defined by:

$$\mu_A(x) = \begin{cases} \dfrac{x-a}{b-a}, & a \leq x \leq b \\ \dfrac{c-x}{c-b}, & b \leq x \leq c \\ 0, & otherwise \end{cases}$$

Step 2.2 Calculating solution architecture value. The contribution of selected decision alternatives, to a specific goal *g* in a given architecture $arch \in AS$ is the cumulative summation of selected alternatives' contributions to the goal. It is defined through equation (i):

$$\widetilde{S_g}(arch) = \textstyle\sum_{a\, \in arch}(\widetilde{con_{g,a}} x_a) \ \text{(i)}$$

In (i), for each decision alternative $a \in A$, we consider a binary decision variable x_a, indicating whether an alternative is chosen, i.e. x_a=1, or not chosen, i.e. x_a=0. Typically, stakeholders may have different emphasises on the expected system quality goals. For each goal $g \in G$, we assign a numeric value between $P_g \in$ [1..10] that shows the degree of priority of the goal in view of stakeholders. Goals in a big data adoption scenario can be divided to two groups as G_{min} and G_{max} in the sense that some goals are supposed to be either maximized e.g. scalability or minimized e.g. cost. Therefore, the total value of a candidate solution architecture $arch \in AS$ is the subtracting the total value of goals that should be maximized from those that are expected to be minimized. This is defined via equation (ii) which gets a triangular fuzzy number:

$$\widetilde{S}\,(arch) = \textstyle\sum_{g\, \in G_{max}} (P_{g\widetilde{S_g}(arch)}) \ - \sum_{g\, \in G_{min}} (P_{g\widetilde{S_g}(arch)}) \ \text{(ii)}$$

Our aim is to find an $arch \in AS$ with highest value of $\tilde{S}(arch)$.

Step 2.3. Specifying solution architecture constraints. A goal may have a certain constraint that has to be satisfied, e.g. the constraint for *Achieve [Reduced processing sensor data]* is expected to be less than 40 millisecond. A constraint for goal *g*, represented by Crt_g, defined through equation (iii):

$$\forall g \in G_{max} : Crt_g \leq \widetilde{S_g}(arch) \ \text{(iii)}$$

$$\forall g \in G_{min} : \widetilde{S_g}(arch) \leq Crt_g$$

The equation (iii) can be simply re-written for g_4. In accepting or rejecting a solution architecture, its cost is also an important constraint, which is shown using Crt_c and is defined using equation (iv):

$$\sum_{a\, \epsilon arch}(\widetilde{cost}_a x_a) \leq \ Crt_c \quad \text{(iv)}$$

$\widetilde{cost}_a$ shows the fuzzy cost of the decision alternative a. Constraints on goals and cost are defined regarding project context in which they are applied.

Step 2.4. Comparing solution architectures. Given the obtained values for $\widetilde{S}$(arch), we can compare architectures to find appropriate one. Between two architectures arch$_1$ and arch$_2$ $\in$ AS , arch$_1$ is more desirable if: $\widetilde{S}\,(arch_1) \ \leq \widetilde{S}\,(arch_2)$ .This is a fuzzy comparison of two fuzzy ranges of possible values for two architectures resulting in the one with a better range. For this, we employ Chen's method (Chen 1985) where it defines the concepts of fuzzy maximizing and minimizing sets expressed using equations (v) and (vi):

$$S_{max}(x)\text{= } (x - x_{min}/x_{max} - x_{min})^k \quad \text{(v)}$$

$$S_{min}(x)\text{= } (x_{max} - x/x_{max} - x_{min})^k \quad \text{(vi)}$$

In equations (v) and (vi), S_{max} = sup $\bigcup_{i=1}^{n} sup\, S_i$ and S_{min} = inf $\bigcup_{i=1}^{n} sup\, S_i$ and k > 0 is a real number. Using these two sets, left and right utility of a fuzzy number S_i , i^{th} architecture solution, is defined as:

$$\text{L } (\widetilde{S}\,(arch_i)\text{= sup min } (S_{min}(x), \widetilde{S}\,(arch_i)) \quad x \in \mathbb{R}$$

$$\text{R } (\widetilde{S}\,(arch_i)) \text{ = sup min } (S_{max}(x), \widetilde{S}\,(arch_i)) \quad x \in \mathbb{R}$$

Given that, the ranking index for i^{th} solution architecture is obtained using the equation (vii):

$$CH^k(\widetilde{S}\,(arch_i)) = ½ \text{ (R } (\widetilde{S}\,(arch_i)) + 1 \text{ - L } (\widetilde{S}\,(arch_i)) \quad \text{(vii)}$$

Step 2.5. Finding the optimum solution architecture. Finding an architecture optimizing quality goals is a typical multi-objective optimization problem under some constraints (Zimmermann 1978). A solution architecture is considered optimal if it maximizes quality goals and satisfies imposed constraints which is, in fact, defined as a linear programming problem through equation (viii):

$$\text{Maximize } \widetilde{S}\,(arch) \text{ subject to the constraint equations (iii) and (iv)} \qquad \text{(viii)}$$

(viii) maximizes the cumulative value of architectures by selecting a combination of alternatives resulting in the highest architecture value under equations (iii) and (iv) to avoid constraint violation.

4 Application example

Our scenario of integrating HMCS with multiple big data analytics platforms (see Section 2.1) is used to explore the architectural solution space. High-level goals for reengineering HMCS to big data analytics platforms were elaborated into potential obstacles, resolution tactics and operational decision alternatives (Figures 2 - 6). Tables 4 shows how the alternatives and their resolution tactic form a space of possible solution architectures.

Table 4. Goals and decision alternatives

Goal	Decision	Alternative
G1. Achieve [Processed social media weekly under expected time]	d0. Substitute goal	Not applicable (the goal definition is refined)
	d1. Social media data processing	Python NLTK Gate Lexalytics Sentiment Toolkit AeroText
	d2.Scheduler	Fair scheduler Capacity scheduler Delay scheduler Matchmaking scheduler
G2. Achieve [Processed sensor data under expected time]	d6. Reduce obstacle Big data analytic platform latency	Acquire more resources
	d7. Reduce obstacle Performance variability of big data platform	Refine network topology
	d8. Reduce obstacle Sensor data exceeds expected	Not applicable

	processing time	
	d9. Reduce obstacle Sensor data exceeds expected processing time	Prioritizing sensor data processing (a24)
	d3. Real-time stream processing	SQLStream Storm StreamCloud
	d10. Restore goal for obstacle Cloud transient fault	Retry connection
G3. Achieve [Improved availability]	d11. Restore goal	Eventual Consistency Weak Consistency Timeline Consistency
G4. Achieve [Maintained interoperability with other big data platforms]	d12. Prevent obstacle	Adapt data
	d13.Prevent obstacle	Develop adaptor
G5. Achieve [Improved data visualization]	d4. Data visualization	Google chart Tableau Data-driven document (d3) Fusion chart
G6. Achieve [Maintained security of sensor data]	d14.Prevent obstacle	Redact data Mask data Obfuscate data
G7. Achieve [Increased unstructured data storage capacity]	d5. Big data storage	MongoDB Accumulo HBase Cloudant BigTable

Through collaboration with stakeholders, the system architect specifies the impact of decision alternatives on the goals. To this end, she used linguistic variables shown in Table 5 and triangular fuzzy membership functions, defined in Step 2, for the variables (Figure 7).

The complexity of the selecting decision alternatives to generate a proper solution architecture in the goal model (Figure 6) is revealed when the system architecture faces with a large number of goals and decision alternatives. The scenario integrating HMCS with big analytics data platforms follows 9 goals that are expected to be satisfied. Table 6 shows 29 different decision alternatives in total. Considering that for each decision only one alternative can be selected, there are 5*5*4*4*3*3*1*1*3 = 10800 potential alternative architectural solutions, each of which represents a trade-off among the goals. Predicting the impact of potential solution architectures on the goals and finding which portion of the architectural space is valid can be a challenging exercise, in particular at the early stage of the system big data enablement process where the real impact of decision alternatives on quality goals is uncertain. Using the second step of the approach (Section 3.2), the system architect is able to explore the space of solution architectures for HMCS.

Table 5 Linguistic variables used to show the impact of decision alternatives on quality goals

Linguistic term	α level cuts	
	1-level cut	0-level cut
Very low (VL)	0	0.25
Low (L)	0.25	0, 0.5
Medium (M)	0.5	0.25, 0.5
High (H)	0.75	0.5, 1
Very high (VH)	1	0.75

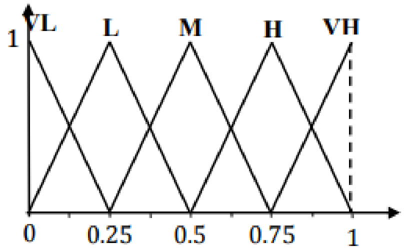

Figure 7. Membership function of the linguistic variables

Table 6. Impact of decision alternatives on expected system quality goals (NA: Not applicable)

		a) The fuzzy expression of decision alternatives' impact on goals						b) Simple crisp expression of decision alternatives' impact on goals (values between 1 and 5)					
Decision	**Alternative**	**Goal**						**Goal**					
		G1	**G2**	**G3**	**G4**	**G5**	**G6**	**G1**	**G2**	**G3**	**G4**	**G5**	**G6**
d0.Substiute goal	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
d1. Social media data processing	Python NLTK	L	VL	H	L	L	H	2.7	1.8	4.3	2.4	2.2	4
	Gate	M	VL	M	M	M	VH	3.5	1.3	3.7	3.9	3.6	5
	Lexalytics Sentiment Toolkit	M	M	L	M	L	L	3.3	3.8	2.8	3.2	2.8	2
	AeroText	VH	L	H	L	H	L	5.9	2.4	4.4	2.1	4.1	2
d2.Scheduler	Fair scheduler (a1)	H	M	VH	M	VH	L	4.2	3.9	5	3.6	5	2
	Capacity scheduler (a2)	VL	VL	M	M	L	L	1.3	1.2	3.2	3.4	2.8	2
	Delay scheduler (a3)	M	VL	M	M	VH	M	3.6	1.8	3.7	3.9	5	3
	Matchmaking scheduler (a4)	M	H	H	M	H	L	3.2	4.3	4.3	3.3	4.3	2
d3.Real-time stream processing	SQLStream (a11)	VL	H	M	H	L	VH	1.8	4.6	3.9	4.1	2.1	5
	Storm (a12)	M	VL	M	H	L	VL	3.7	1.3	3.7	4.3	2.5	1
	StreamCloud (a13)	M	M	VH	L	M	L	3.3	3.6	5	2.7	3.2	2
d4. Data visualization	Google chart (a6)	L	M	H	M	VH	VL	2.2	3.1	4.2	3.6	5.2	1.7
	Tableau (a7)	L	M	H	VH	M	VL	2.3	3.7	4.6	5	3.2	1.5
	Data-driven (a8)	M	M	M	VH	M	VL	3.2	3.8	3.7	5	3.3	1.9
	Document (d3) (a9)	H	M	M	M	H	H	4.1	3.1	3.2	3.6	4.5	4.9
	Fusion chart (a10)	VL	H	M	M	L	VH	1.7	4.3	3.8	4.3	2.4	5
d5. Big data store	MongoDB (a1)	L	VL	M	H	VH	VH	2.1	1.7	3.3	4.5	5.4	5
	Accumulo (a2)	L	L	M	M	VH	H	2.6	2.8	3.3	3.3	5	4.2
	HBase (a3)	H	L	H	M	VH	VH	5	2.6	4.3	3.2	5	5
	Cloudant (a4)	M	H	M	VL	M	VL	3.9	4.6	3.7	1.6	3.2	1.9
	BigTable (a5)	M	M	M	H	H	VH	3.4	3.7	3.6	4.4	4.7	5
d6. Reduce obstacle	Acquire more resources	M	M	M	M	VH	H	3.8	3.8	3.2	3.1	5	4
d7. Reduce obstacle	Refine network topology	VL	H	M	M	L	VH	1.7	4.3	3.8	4.3	2.4	5

d8 – d10. Reduce obstacle	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA	NA
d11. Restore goal	Eventual consistency (a27)	VH	M	L	H	VH	H	5	3.3	1.7	4.7	5	4
	Weak consistency (a28)	VL	M	M	M	M	M	1.3	3.6	3.9	3.9	3.2	3
	Timeline consistency (a29)	M	VH	M	M	VH	H	3.8	5	3.2	3.4	5	4
d12. Prevent obstacle	Adapt data (a25)	VL	M	M	H	H	M	1.4	3.9	3.5	4.5	4.8	3
d13. Prevent obstacle	Develop adaptor (a26)	M	L	VL	M	VL	M	3.6	2.2	1.6	3.6	1.4	3
d14. Prevent obstacle	Redact data (a22)	M	VL	L	M	M	M	3.2	1.8	2.4	3.9	3.6	3
	Mask data (a23)	M	M	M	M	VH	H	3.8	3.8	3.2	3.1	5	4
	Obfuscate data (a24)	VL	H	L	M	M	H	1.1	4.2	2.8	3.9	3.6	4

Given that the goal weights assumed equal by stakeholders, the value of each solution architecture is calculated using equation (ii). One of the constraints imposed by stakeholders is to keep the cost of a solution architecture under $1000. This constraint is represented using equation (iv). Step 2 shows that this constraint rules out 2531 solution architectures (of all possible 10800). Following further discussions with stakeholders, it is agreed to relax this constraint to $1200. This reduces the number of rejected architectures from 2531 to 1652. As mentioned in Section 3.2, the second constraint imposed by users of HMCS is to keep the data stream processing coming from sensors below 40 milliseconds. This allows the system architect to assess the system performance constraint on the choice of solution architectures. Relaxing this constraint to 47 milliseconds yields further increases the number of acceptable candidate architecture solutions. With this change, 185 solution architectures are further rejected (out of 9718). These remaining are further explored. Table 7 shows the top 20 solution architectures ranked using equation (vii) and the selected architecture decision alternatives for all those 20. Recall from Step 2.4 (Section 3.2), among two solution architectures the one is better if it has a greater fuzzy value. The best solution architecture ranked as the first one has the best combination of decision alternatives in view of trade-off among goals compared to majority of candidates. That is, it has the best combination of fuzzy values. Nevertheless, it is still likely that the architect may select a solution architecture that is slightly worse than the optimal one due to some reasons not captured by our approach. For example, the system architect may prefer a particular big data analytics platform in the marketplace. Figure 8 shows the final goal model based on the selected decision alternatives for the first ranked solution architecture using fuzzy logic.

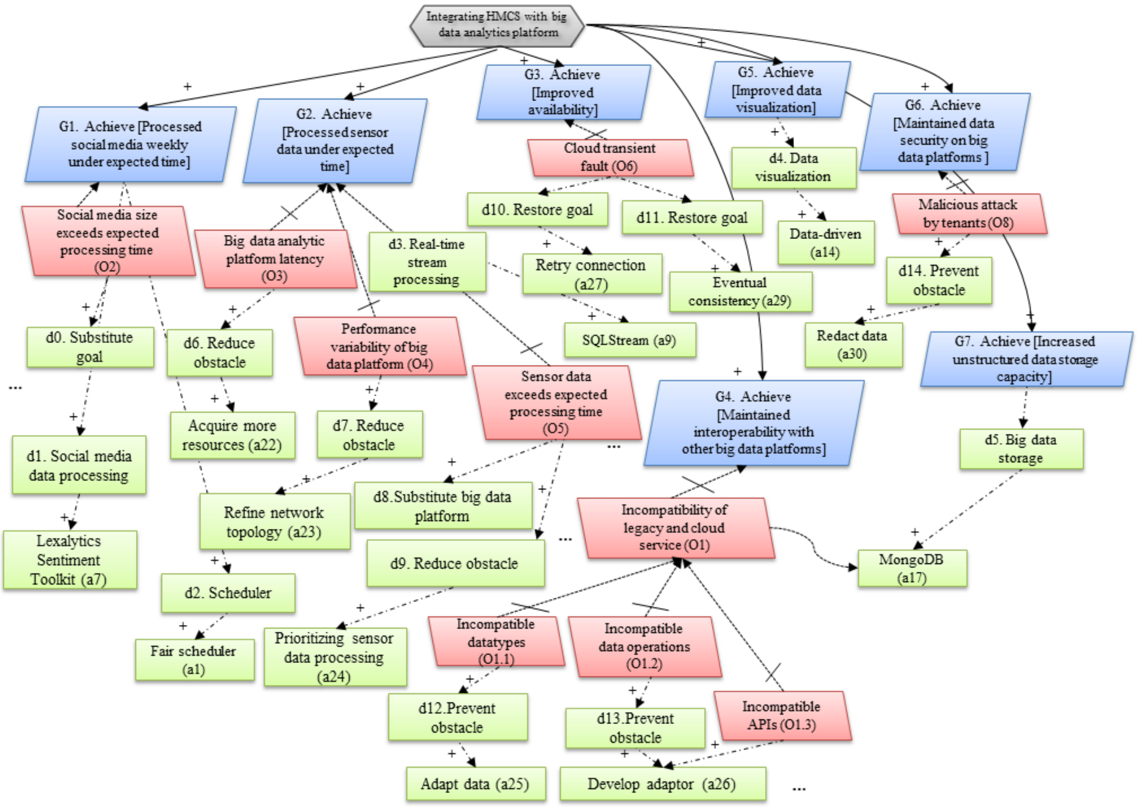
Integrating HMCS with big data analytics platform
G1. Achieve [Processed social media weekly under expected time]
G2. Achieve [Processed sensor data under expected time]
G3. Achieve [Improved availability]
G5. Achieve [Improved data visualization]
G6. Achieve [Maintained data security on big data platforms]
Cloud transient fault (O6)
d4. Data visualization
Social media size exceeds expected processing time (O2)
Big data analytic platform latency (O3)
d3. Real-time stream processing
d10. Restore goal
d11. Restore goal
Data-driven (a14)
Malicious attack by tenants (O8)
Retry connection (a27)
Eventual consistency (a29)
d14. Prevent obstacle
d0. Substitute goal
d6. Reduce obstacle
Performance variability of big data platform (O4)
SQLStream (a9)
Redact data (a30)
G7. Achieve [Increased unstructured data storage capacity]
...
Sensor data exceeds expected processing time (O5)
G4. Achieve [Maintained interoperability with other big data platforms]
Acquire more resources (a22)
d7. Reduce obstacle
d1. Social media data processing
d5. Big data storage
d8.Substitute big data platform
Refine network topology (a23)
Incompatibility of legacy and cloud service (O1)
Lexalytics Sentiment Toolkit (a7)
d9. Reduce obstacle
MongoDB (a17)
d2. Scheduler
Fair scheduler (a1)
Prioritizing sensor data processing (a24)
Incompatible datatypes (O1.1)
Incompatible data operations (O1.2)
Incompatible APIs (O1.3)
d12.Prevent obstacle
d13.Prevent obstacle
Adapt data (a25)
Develop adaptor (a26)
...

Figure 8. Selected decision alternatives for the first solution architecture including the best combination of decision alternatives

Table 7. Ranking solution architectures with respect to the decision alternatives based on fuzzy-logic and crisp approaches

	(a) Fuzzy approach												(b) Crisp approach												
	Decision alternative												Decision alternative												
	d1.Social media data processing	d2.Scheduler	d3.Real-time stream processing	d4.Data visualization	d5.Data store	d6. Reduce obstacle	d7. Reduce obstacle	d8-d10	d11. Restore goal	d12. Prevent obstacle	d13. Prevent obstacle	d14. Prevent obstacle		d1.Social media data processing	d2. Scheduler	d3.Real-time stream processing	d4.Data visualization	d5.Data store	d6. Reduce obstacle	d7. Reduce obstacle	d8-d10	d11. Restore goal	d12. Prevent obstacle	d13. Prevent obstacle	d14. Prevent obstacle
Rank													Rank												
1	Lexalytics Sentiment Toolkit	Fair scheduler	SQLStream	Data-driven	MongoDB	Acquire more resources	Refine network topology	NA	Eventual Consistency	Adapt data	Develop adaptor	Redact data	1	Gate	Fair scheduler	Storm	Google chart	Accumulo	Acquire more resources	Refine network topology	NA	Timeline Consistency	Adapt data	Develop adaptor	Obfuscate data
2	Gate	Capacity scheduler	StreamCloud	Document	Cloudant	Acquire more resources	Refine network topology	NA	Timeline Consistency	Adapt data	Develop adaptor	Mask data	2	AeroText	Capacity scheduler	StreamCloud	Document	BigTable	Acquire more resources	Refine network topology	NA	Eventual Consistency	Adapt data	Develop adaptor	Redact data
3	Gate	Capacity scheduler	StreamCloud	Fusion chart	Cloudant	Acquire more resources	Refine network topology	NA	Timeline Consistency	Adapt data	Develop adaptor	Obfuscate data	3	Python NLTK	Matchmaking scheduler	SQLStream	Tableau	Cloudant	Acquire more resources	Refine network topology	NA	Timeline Consistency	Adapt data	Develop adaptor	Mask data
4	Lexalytics Sentiment Toolkit	Matchmaking scheduler	SQLStream	Data-driven	Accumulo	Acquire more resources	Refine network topology	NA	Weak Consistency	Adapt data	Develop adaptor	Redact data	4	Gate	Fair scheduler	StreamCloud	Data-driven	MongoDB	Acquire more resources	Refine network topology	NA	Weak Consistency	Adapt data	Develop adaptor	Mask data
5	Lexalytics Sentiment Toolkit	Fair scheduler	Storm	Fusion chart	Tableau	Acquire more resources	Refine network topology	NA	Eventual Consistency	Adapt data	Develop adaptor	Mask data	5	Lexalytics Sentiment Toolkit	Fair scheduler	SQLStream	Google charte	BigTable	Acquire more resources	Refine network topology	NA	Eventual Consistency	Adapt data	Develop adaptor	Obfuscate data
6	Gate	Delay scheduler	Storm	Data-driven	Cloudant	Acquire more resources	Refine network topology	NA	Timeline Consistency	Adapt data	Develop adaptor	Obfuscate data	6	Gate	Capacity scheduler	Storm	Fusion chart	Cloudant	Acquire more resources	Refine network topology	NA	Weak Consistency	Adapt data	Develop adaptor	Redact data
7	AeroText	Matchmaking scheduler	StreamCloud	Python NLTK	HBase	Acquire more resources	Refine network topology	NA	Weak Consistency	Adapt data	Develop adaptor	Redact data	7	Gate	Delay scheduler	StreamCloud	Document	HBase	Acquire more resources	Refine network topology	NA	Weak Consistency	Adapt data	Develop adaptor	Mask data
8	Lexalytics Sentiment Toolkit	Capacity scheduler	Storm	Document	Google chart	Acquire more resources	Refine network topology	NA	Weak Consistency	Adapt data	Develop adaptor	Obfuscate data	8	AeroText	Capacity scheduler	SQLStream	Tableau	Cloudant	Acquire more resources	Refine network topology	NA	Timeline Consistency	Adapt data	Develop adaptor	Mask data
9	Gate	Fair scheduler	SQLStream	Fusion chart	MongoDB	Acquire more resources	Refine network topology	NA	Timeline Consistency	Adapt data	Develop adaptor	Redact data	9	Lexalytics Sentiment Toolkit	Matchmaking scheduler	StreamCloud	Fusion chart	Cloudant	Acquire more resources	Refine network topology	NA	Timeline Consistency	Adapt data	Develop adaptor	Obfuscate data
10	AeroText	Delay scheduler	Storm	Python NLTK	BigTable	Acquire more resources	Refine network topology	NA	Eventual Consistency	Adapt data	Develop adaptor	Obfuscate data	10	Lexalytics Sentiment Toolkit	Delay scheduler	SQLStream	Data-driven	MongoDB	Acquire more resources	Refine network topology	NA	Eventual Consistency	Adapt data	Develop adaptor	Obfuscate data

Interestingly, the system architect also compares the results generated through our approach and a simple crisp approach (Table 7-b). She uses crisp values to represent the impact of decision alternatives on the goals. Note that by the simple crisp approach, we mean all approaches ignoring the uncertainty in the impact of decision alternatives on quality goals. This difference highlights the contribution of our approach compared to crisp approach in the selection of a proper solution architecture. The simple crisp approach for the calculation the value of a candidate solution architecture would select 125th solution architecture as the optimal solution. This is in contrast to our fuzzy approach in which 125th approach is ranked as 46th suitable candidate architecture. In other words, 125th has a large negative consequence of uncertainty, which is ignored by the crisp approach. The difference between 1th and 125th solution architectures can be also recognized through specific decision alternatives selected for each architecture. Table 7 represents the selected decision alternatives for the first top solution architectures based on fuzzy logic and crisp approaches. For example, regarding the information in this right and left sides of the table to find selected decision alternatives, it is observable that our approach chose *MongoDB* for *Data Store* for the first ranked solution architecture whilst the crisp approach chose *Accumulo*. Figure 8 shows the final goal model based on the selected decision alternatives for the first ranked solution architecture using fuzzy logic.

5 Related work

There is a paucity of research focus on the early goal-obstacle analysis and architecture decisions in the scope of integrating manufacturing systems with big data analytics platforms. The literature most related is thus subsumed under three research streams: (i) traditional system re-engineering, (ii) reengineering to cloud platforms, and (iii) reengineering to big data analytics platforms. Hence, we discuss how our approach presented is positioned in relation to notable research in each stream.

5.1 Traditional approaches for legacy system reengineering

Early decision making on selecting solution architectural under uncertainty have been already discussed in the Introduction section. One of the earliest work is by Svahnberg et al. where they provide a multi-criteria decision method using Analytic Hierarchy Process (AHP) supporting comparison of different software architecture candidates for software quality attributes (Svahnberg, Wohlin, Lundberg et al. 2003). In its process, two sets of vectors of different candidate architectures with respect to different quality attributes and vice versa are created and refined. The variance of uncertainty is also calculated in each candidate architecture. The sets are used as input to a consensus decision making process in a way that disagreements among stakeholders are discussed to get understanding reasons behind. Our proposed equations in this work are inspired by GuideArch approach (Esfahani, Malek and Razavi 2013). It presents a fuzzy-based exploration of the architectural solution space under uncertainty aiding architecture in making architecture selection. GuideArch is later extended in (Letier, Stefan and Barr 2014) where authors model uncertainty about parameters' values as probability distributions rather than fuzzy values and also assess to extent additional information about uncertain parameters can reduce risks. All of these works including others e.g. (Al-Naeem, Gorton, Babar et al. 2005) do not provide a systematic support for top-down goal-obstacle analysis towards generating possible decision architecture alternatives in view of system quality goals. Our approach can be used as a complementary step to generate different alternatives to be used as an input for this group of studies to identify suitable solution architecture.

5.2 Legacy systems and cloud computing

An impetus to look at this track of research is the popularity of hosting big data solution architectures on the cloud computing platforms (Agrawal, Das and El Abbadi 2011). Khajeh□Hosseini et al. (Khajeh□Hosseini, Greenwood, Smith et al. 2012) define a cloud adoption conceptual framework to support decision makers in identifying uncertainties. They focus particularly on the cost of deploying options of legacy systems in cloud platforms, which may undergo network latency and service price. Coth studies limit their view to the cost of legacy system reengineering. Similarly, Umar et al. (Umar and Zordan 2009) defines decision model for reengineering legacy systems to service-oriented architecture to make trade-off between integration versus migration in terms of cost. On the contrary, We do not confine our view to the reengineering cost; rather incorporate other system quality goals that might be important for stakeholders along with elaborating them to potential obstacles and operationalization alternatives. The approach in (Zardari, Bahsoon and Ekárt 2014) uses goal-obstacle analysis to represent risks encountered in using cloud services and mitigating strategies. We extended Zardari's goal-oriented approach by taking into account the risk of uncertainty impact of decision alternatives on stakeholders' goals using fuzzy math. Furthermore, previous cloud migration literature (Alonso, Orue-Echevarria, Escalante et al. 2013), (Menzel, Schönherr and Tai 2013), (Menzel and Ranjan 2012), and (Zimmermann 2017) suffer providing a meticulous process for an early goal-obstacle exploration and architecture decisions with considering uncertainty issue at the same time.

5.3 Legacy systems and big data

At the organizational level, some studies have strived in identifying and analyzing business goals in the early stage of big data adoption to get better addressing stakeholders' concerns. For example, Park emphasizes the importance of alignment between organizational business processes and big data sides towards making better business decisions to adopt big data analytics platforms (Park 2017). She defines a systematic process to ensure traceability among high-level big data adoption goals and big data solutions in view of relevance (utility of a data element), comprehensiveness (preventing omissions of potentially important data), and prioritization (required effort in obtaining resources for the data). In another work, Supakkul's approach discusses insights gained from adopting big data to improve business goals (Supakkul, Zhao and Chung 2016). Their approach generates two types of resulting insight through goal reasoning and decision-making: (i) descriptive insights of current state of business e.g. the customer retention rate and (ii) predictive insights e.g. customers who are likely to defect. GOBIA (Goal-Oriented Business Intelligence Architecture) is a goal-oriented approach for transforming business goals into a customized big data architecture (Fekete 2016). GOBIA produces a layered-based conceptual solution architecture, which can be realized by selecting an appropriate mix of big data analytics platforms, though it leaves technology selection to system implement phase. The main difference between our approach and the above studies is that we narrow our focus on legacy systems as the unit of analysis and explore solution architectures to integrate them with big data analytics platforms.

On the other hand, some work deal with integrating existing legacy systems with big data analytics platforms. This genre of literature is deemed closest to our work. A key feature of existing works is their motivations in making legacy systems big data enablement. Some studies develop intelligent techniques such as clustering (Fahad, Alshatri, Tari et al. 2014), deep learning (Najafabadi, Villanustre, Khoshgoftaar et al. 2015), text mining (Xiang, Schwartz, Gerdes et al. 2015), and machine learning algorithms (Scott, Blocker, Bonassi et al. 2016) on big data analytic platforms to mine hiding knowledge in given legacy system data. Once chosen, such techniques can supply inputs to the second step of our approach as decision alternatives for the goal operationalization or obstacle resolution (see third column of Table 4 for example) where their impact on quality goals is investigated for the optimum selection of solution architecture.

Jha et al. define both forward and backward reengineering activities through which legacy system functionalities are reused, and their data can be accessed and processed by big data analytics platforms (Jha, Jha, O'Brien et al. 2014). They suggest a framework to construct an architectural view of big data solution including business, data, and application architecture (Jha, Jha and O'Brien 2015). The need for this is highlighted by (Varkhedi, Thati, Nanda et al. 2014) discussing challenges of transferring legacy system data, e.g. mainframes, to a platform configured for big data processing in the same or a separate logical partition on the legacy systems. (Mathew and Pillai 2015) suggest a three layer-based architecture for handling heterogeneities between legacy systems and big data analytics platforms. Govindarajan's work, as a part of Cloud Collaborative Manufacturing Networks (CCMN) project, resolves integrating supply chain manufacturing system and logistic assets with cloud services by developing data adapters for collecting and transforming data from heterogeneous sources to appropriate format accepted by legacy systems, i.e. XML (Govindarajan, Ferrer, Xu et al. 2016). Similarly, Givehchi et al. provide a cross-layer architecture enabling interoperability between legacy industrial devices (e.g. I/O devices and sensors) and big data analytics platforms (Givehchi, Landsdorf, Simoens et al. 2017). They apply an information model in order to retain legacy device codes unchanged.

While above approaches acknowledge challenges in legacy system big data enablement scenarios, they keep the description of their analysis process at a high-level that does not represent 'actionable intelligence' towards big data enablement. Nor do they address the uncertainty issue. We have prescribed a more detailed approach for analyzing goals in moving manufacturing systems to big data platforms, identifying, assessing, and generating resolution tactics in handling potential risks (i.e. step 1 of the approach). Furthermore, we address selecting, prioritization, and ranking resolution tactics in identifying proper solution big data solution architecture under uncertainty (i.e. step 2 of the approach). We have not found other studies that outline a systematic approach on early requirements and big data architecture decisions.

Giret et al. state that the main reason of complexity for developing service-oriented manufacturing systems is the number of heterogeneous technologies and execution environments (Giret, Garcia and Botti 2016). They combine multi-agent system design with service-oriented architectures for the development of intelligent automation control and execution of manufacturing systems. Giret later proposes a process model, named Go-green, including activities, guidelines, and tools to design and develop sustainable manufacturing system architectures (Giret, Trentesaux, Salido et al. 2017). We believe that the second step of our approach can augment the design phase of Giret' work to fill its gap in addressing early architecture design of big data enabled manufacturing systems under the uncertainty.

6 Conclusion, limitations and further work

Current manufacturing systems are moving to the next generation, namely Industry 4.0. It is expected that they become widely able to utilize big data analytics platforms for advanced information analytics. However, at this early stage, a clear understanding of goals and risks against big data adoption and how they relate to manufacturing systems is particularly crucial. As business risk management strategy, a system approach to re-architecting existing manufacturing systems is an important contribution. Our goal-obstacle analysis which takes into account imperfect information and unavoidable uncertainties is quite intuitive to follow. In particular, it provides an early stage analysis, which is taken place before delving into technical aspects of big data analytics platforms.

We proposed an approach applies goal-oriented modelling and fuzzy-based logic for analysing suitability of big data solution architecture for manufacturing systems. The approach starts with identifying high-level big data adoption goals, architectural decision alternatives to realize these goals and probable obstacles and analysing uncertainties in selecting solution architecture. The output of the approach is a model relating big data adoption goals to alternatives solution architectures. It gives the system architects a complete set of requirements to be incorporated into the implementation stage to make appropriate trade-offs based on, for instance, cost, security, or performance goals. We also demonstrated the in a scenario of moving a legacy HMCS to a set of big data platforms. This is the first attempt at providing a systematic approach for the early stage of analysis of big data solution architecture. Our proposed approach is novel in that it employs goal reasoning and fuzzy logic to support designing big data solution architecture for manufacturing systems. To the best of our knowledge, such a harness is not available in the literature.

Although we have shown the applicability of our approach, further validation is required to account for the variety in scenarios of integrating manufacturing systems with big data analytics platforms. In other words, there might be some other ways to satisfy goals or some hidden factors that hinder certain goal achievement but are not detected in the approach's steps. Another important way for the improvement of the approach is to provide further automatic support. The size of the goal model in Step 1 and the number of required computations in Step 2 can limit the usability of the framework without further supporting tools. We plan to provide a tool support that facilitates using the approach when working with large-scale goal models.

References

Agrawal, D., S. Das and A. El Abbadi (2011). Big data and cloud computing: current state and future opportunities. Proceedings of the 14th International Conference on Extending Database Technology, ACM.

Al-Naeem, T., I. Gorton, M. A. Babar, et al. (2005). A quality-driven systematic approach for architecting distributed software applications. Proceedings of the 27th international conference on Software engineering, ACM.

Alonso, J., L. Orue-Echevarria, M. Escalante, et al. (2013). Cloud modernization assessment framework: Analyzing the impact of a potential migration to Cloud. Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA), 2013 IEEE 7th International Symposium on the.

Bi, Z. and D. Cochran (2014). "Big data analytics with applications." Journal of Management Analytics **1**(4): 249-265.

Camarinha-Matos, L. M., H. Afsarmanesh, N. Galeano, et al. (2009). "Collaborative networked organizations–Concepts and practice in manufacturing enterprises." Computers & Industrial Engineering **57**(1): 46-60.

Chen, C. P. and C.-Y. Zhang (2014). "Data-intensive applications, challenges, techniques and technologies: A survey on Big Data." Information Sciences **275**: 314-347.

Chen, S.-H. (1985). "Ranking fuzzy numbers with maximizing set and minimizing set." Fuzzy Sets and Systems **17**(2): 113-129.

Dubey, R., A. Gunasekaran, S. J. Childe, et al. (2016). "The impact of big data on world-class sustainable manufacturing." The International Journal of Advanced Manufacturing Technology **84**(1-4): 631-645.

Esfahani, N., S. Malek and K. Razavi (2013). GuideArch: guiding the exploration of architectural solution space under uncertainty. Software Engineering (ICSE), 2013 35th International Conference on, IEEE.

Fahad, A., N. Alshatri, Z. Tari, et al. (2014). "A survey of clustering algorithms for big data: Taxonomy and empirical analysis." IEEE transactions on emerging topics in computing **2**(3): 267-279.

Fekete, D. (2016). The GOBIA Method: Fusing Data Warehouses and Big Data in a Goal-Oriented BI Architecture. GvD.

Franklin, C. (1996). "Lt. Gen (USAF) Commander ESC, January 1996, Memorandum for ESC Program Managers, ESC/CC." Risk Management, Department of the Air Force, Headquarters ESC (AFMC) Hanscom Air Force Base, MA.

Gartner (2015). Gartner Says Business Intelligence and Analytics Leaders Must Focus on Mindsets and Culture to Kick Start Advanced Analytics, availalbe at http://www.gartner.com/newsroom/id/3130017.

Giret, A., E. Garcia and V. Botti (2016). "An engineering framework for service-oriented intelligent manufacturing systems." Computers in Industry **81**: 116-127.

Giret, A., D. Trentesaux, M. A. Salido, et al. (2017). "A holonic multi-agent methodology to design sustainable intelligent manufacturing control systems." Journal of Cleaner Production.

Givehchi, O., K. Landsdorf, P. Simoens, et al. (2017). "Interoperability for industrial cyber-physical systems: an approach for legacy systems." IEEE Transactions on Industrial Informatics.

Govindarajan, N., B. R. Ferrer, X. Xu, et al. (2016). An approach for integrating legacy systems in the manufacturing industry. Industrial Informatics (INDIN), 2016 IEEE 14th International Conference on, IEEE.

Jha, M., S. Jha and L. O'Brien (2015). Integrating big data solutions into enterprize architecture: constructing the entire information landscape. The International Conference on Big Data, Internet of Things, and Zero-Size Intelligence (BIZ2015).

Jha, S., M. Jha, L. O'Brien, et al. (2014). Integrating legacy system into big data solutions: Time to make the change. Computer Science and Engineering (APWC on CSE), 2014 Asia-Pacific World Congress on, IEEE.

Khajeh-Hosseini, A., D. Greenwood, J. W. Smith, et al. (2012). "The cloud adoption toolkit: supporting cloud adoption decisions in the enterprise." Software: Practice and Experience **42**(4): 447-465.

Lee, J., H.-A. Kao and S. Yang (2014). "Service innovation and smart analytics for industry 4.0 and big data environment." Procedia Cirp **16**: 3-8.

Letier, E. (2001). Reasoning about agents in goal-oriented requirements engineering, PhD thesis, Université catholique de Louvain.

Letier, E., D. Stefan and E. T. Barr (2014). Uncertainty, risk, and information value in software requirements and architecture. Proceedings of the 36th International Conference on Software Engineering, ACM.

Letier, E. and A. Van Lamsweerde (2004). Reasoning about partial goal satisfaction for requirements and design engineering. ACM SIGSOFT Software Engineering Notes, ACM.

Li, J., F. Tao, Y. Cheng, et al. (2015). "Big data in product lifecycle management." The International Journal of Advanced Manufacturing Technology **81**(1-4): 667-684.

Lim, S. L. and A. Finkelstein (2012). "StakeRare: using social networks and collaborative filtering for large-scale requirements elicitation." Software Engineering, IEEE Transactions on **38**(3): 707-735.

Lin, H.-K., J. A. Harding and C.-I. Chen (2016). "A Hyperconnected Manufacturing Collaboration System Using the Semantic Web and Hadoop Ecosystem System." Procedia Cirp **52**: 18-23.

Lycett, M. (2013). 'Datafication': Making sense of (big) data in a complex world, Springer.

Mathew, P. S. and A. S. Pillai (2015). Big Data solutions in Healthcare: Problems and perspectives. Innovations in Information, Embedded and Communication Systems (ICIIECS), 2015 International Conference on, IEEE.

McAfee, A., E. Brynjolfsson and T. H. Davenport (2012). "Big data: the management revolution." Harvard Business Review **90**(10): 60-68.

Menzel, M. and R. Ranjan (2012). CloudGenius: decision support for web server cloud migration. Proceedings of the 21st international conference on World Wide Web, ACM.

Menzel, M., M. Schönherr and S. Tai (2013). "(MC2)2: criteria, requirements and a software prototype for Cloud infrastructure decisions." Software: Practice and Experience **43**(11): 1283-1297.

Najafabadi, M. M., F. Villanustre, T. M. Khoshgoftaar, et al. (2015). "Deep learning applications and challenges in big data analytics." Journal of Big Data **2**(1): 1.

Pal, S. K., S. K. Meher and A. Skowron (2015). "Data science, big data and granular mining." Pattern Recognition Letters **67**: 109-112.

Park, E. (2017). IRIS: a goal-oriented big data business analytics framework.

Pedrycz, W. (1994). "Why triangular membership functions?" Fuzzy Sets and Systems **64**(1): 21-30.

Protiviti (2017). Big Data Adoption in Manufacturing: Forging an Edge in a Challenging Environment, Available at: https://www.protiviti.com/sites/default/files/united_states/insights/big-data-adoption-in-manufacturing-protiviti.pdf.

Scott, S. L., A. W. Blocker, F. V. Bonassi, et al. (2016). "Bayes and big data: The consensus Monte Carlo algorithm." International Journal of Management Science and Engineering Management **11**(2): 78-88.

Stark, J. (2015). Product lifecycle management. Product Lifecycle Management (Volume 1), Springer**:** 1-29.

Supakkul, S., L. Zhao and L. Chung (2016). GOMA: Supporting Big Data Analytics with a Goal-Oriented Approach. Big Data (BigData Congress), 2016 IEEE International Congress on, IEEE.

Svahnberg, M., C. Wohlin, L. Lundberg, et al. (2003). "A quality-driven decision-support method for identifying software architecture candidates." International Journal of Software Engineering and Knowledge Engineering **13**(05): 547-573.

Umar, A. and A. Zordan (2009). "Reengineering for service oriented architectures: A strategic decision model for integration versus migration." Journal of Systems and Software **82**(3): 448-462.

Van Lamsweerde, A. (2009). "Requirements engineering: from system goals to UML models to software specifications."

van Lamsweerde, A. and E. Letier (2000). "Handling obstacles in goal-oriented requirements engineering." Software Engineering, IEEE Transactions on **26**(10): 978-1005.

Varkhedi, A., V. Thati, A. Nanda, et al. (2014). System and method for extracting data from legacy data systems to big data platforms, Google Patents.

Waller, M. A. and S. E. Fawcett (2013). "Data science, predictive analytics, and big data: a revolution that will transform supply chain design and management." Journal of Business Logistics **34**(2): 77-84.

Wang, H., Z. Xu, H. Fujita, et al. (2016). "Towards felicitous decision making: An overview on challenges and trends of Big Data." Information Sciences **367**: 747-765.

Wegener, R., & Sinha, V. (2013). "The value of big data: How analytics differentiates winners." http://www.bain.com/publications/articles/the-value-of-big-data.aspx.

Xiang, Z., Z. Schwartz, J. H. Gerdes, et al. (2015). "What can big data and text analytics tell us about hotel guest experience and satisfaction?" International Journal of Hospitality Management **44**: 120-130.

Yu, E. S. K. and J. Mylopoulos (1994). Understanding “Why” in Software Process Modelling, Analysis, and Design. Proceedings of the 16th international conference on Software engineering. Sorrento, Italy, IEEE Computer Society Press**:** 159-168.

Zadeh, L. A., K.-S. Fu and K. Tanaka (2014). Fuzzy sets and their applications to cognitive and decision processes: Proceedings of the us–japan seminar on fuzzy sets and their applications, held at the university of california, berkeley, california, july 1-4, 1974, Academic press.

Zardari, S., R. Bahsoon and A. Ekárt (2014). "Cloud Adoption: Prioritizing Obstacles and Obstacles Resolution Tactics Using AHP."

Zimmermann, H.-J. (1978). "Fuzzy programming and linear programming with several objective functions." Fuzzy Sets and Systems **1**(1): 45-55.

Zimmermann, O. (2017). "Architectural refactoring for the cloud: a decision-centric view on cloud migration." Computing **99**(2): 129-145.