DECISION SUPPORT IN IT SERVICE MANAGEMENT: APPLYING AHP METHODOLOGY TO THE ITIL INCIDENT MANAGEMENT PROCESS

Martin Jantscher
FH JOANNEUM University of Applied Sciences
Graz, Austria
E-mail: martin.jantscher@edu.fh-joanneum.at

Christopher Schwarz
FH JOANNEUM University of Applied Sciences
Graz, Austria
E-mail: christopher.schwarz@fh-joanneum.at

Erwin Zinser[1]
FH JOANNEUM University of Applied Sciences
Graz, Austria
E-mail: erwin.zinser@fh-joanneum.at

ABSTRACT

The order in which service engineers in the field of IT service management decide to resolve incidents is crucial considering the impact it has on the business performance of the IT service provider. A possible solution to reduce negative impacts is to support service engineers by means of a decision support system that calculates priorities for the incidents based on their business impact. Considering these priorities in their decisions, service engineers can help reduce such negative impacts on the IT service provider's business. The aim of this study was to incorporate the Analytical Hierarchy Process (AHP) method into a software tool to prioritize incidents according to the severity of their business impact. As a result, a decision support system called Incident Prioritizer (IP) was developed. It uses the AHP method to calculate priorities for incidents based on typical criteria that are relevant in the field of IT service management to assess the business impact of incidents. These criteria are commonly defined by the management of an IT service provider organization, and reflect the management's understanding of the business impact that is caused by incidents. The results suggest that the AHP method can be successfully applied for the given problem of incident prioritization. The prioritization is facilitated by an AHP decision model that consists of only one level of criteria and considers judgments from human beings as well as data from external information systems.

[1] Corresponding author

Keywords: Analytic Hierarchy Process, IT Service Management, Decision Support System, Incident Prioritizer, ITIL.

1. Introduction and objectives

In the era of cloud computing the management of information technology (IT) infrastructure becomes an ever increasing challenge. Large IT service providers (ISP) in particular have to find ways to manage complex, centralized and large-scale IT infrastructure. The service providers are supported in their endeavor by consulting best practice guidelines on managing IT infrastructure effectively such as the IT infrastructure library (ITIL).

It is a common practice for IT service providers to issue contracts to customers that specify the IT service offered as well as numerous service level targets that the IT service provider must meet. Furthermore, the contract specifies penalties that are due in case the IT service provider fails to meet those service level targets. Moreover, it is common for IT service providers to employ service engineers that are responsible for resolving incidents. Incidents are basically interruptions of IT services which have a negative impact on the business performance of affected customers. These interruptions can have various causes such as hardware failure or software crashes. Large IT service providers face dozens of incidents each day that often occur at the same time.

Typically, service engineers decide in which order they resolve those incidents. These decisions are made without any information on the penalties that are caused by the incidents, even though the order in which service engineers decide to resolve incidents can have a severe negative impact on the business of IT service providers (Beims, 2010). A possible solution to close this information gap is to provide the service engineers with a decision support system that prioritizes incidents based on the severity of their business impact. Thus, the overall objective of this study is to incorporate the Analytical Hierarchy Process (AHP) method into ITIL-compliant software to prioritize incidents according to the severity of their business impact. The AHP is a widely accepted approach for decision making in different domains. Consequently, it is also an adequate approach for the purposeful prioritization of incidents.

The objective of this study was achieved by developing a software tool called Incident Prioritizer (IP). The IP uses the AHP method to calculate priorities for incidents based on well-defined decision criteria. It is a decision support system that supports the aforementioned decision process of service engineers by prioritizing all open incidents according to the severity of their business impact. To assess the business impact of incidents multiple criteria are taken into account. These criteria are commonly defined by the management of an IT service provider. The output of the IP is prioritized incidents. The calculated priorities represent the incidents' business impact. When service engineers consider these priorities in the decision process, it can help reduce negative impact on the business of the IT service provider.

This paper describes how exactly the AHP method is applied in the context of the Incident Prioritizer software. Moreover, this paper will unveil implementation details of

the Incident Prioritizer with emphasis on the technical implementation of the AHP method.

2. Literature review

It is a well-known fact in the field of IT service management (ITSM) that incidents have a negative impact on the business performance of IT service providers. The ITIL guideline thus recommends that IT service providers prioritize their incidents based on their business impact (OGC, 2007). Business impacts are factors like financial losses or the number of customers that are affected by an incident. In addition, Beims (2010) states that the order in which service engineers decide to resolve incidents can also contribute to a negative impact on the business of the IT service provider.

Simulation modeling provides an efficient means in an ITSM context to support decision-making (Orta et al., 2014). Numerous simulation techniques exist to address different fields of application aiming at prominent modules of the ITIL ITSM framework, i.e. service strategy, service design, service transition, and service operation. Among them, state-based process models, discrete event simulation, Petri-net models as well as reliable mathematical simulation techniques are commonly applied (Power, 2002). The common denominator of these approaches turns out to be performing prediction with respect to system behavior in order to improve the overall performance of ITIL-related processes. However, many of these simulation experiments rely on synthetic data that is generated by simulation software and, thus, they do not necessarily reflect real-world scenarios. Furthermore, until recently, AHP was not yet extensively used to support decision-making regarding ITSM. Repschlaeger et al. (2014) successfully applied AHP for provider selection of a cloud-based IT service management system. Wan et al. (2011) determined various indicators of IT service management process where weights of each indicator were calculated by the AHP methodology. Moreover, Wan and Wan (2012) made use of AHP in order to improve the overall performance of global companies with respect to their IT service management processes.

Incident management as part of the service operation module of the ITIL framework deals with managing and restoring normal service operation after an interruption to minimize severe impacts on the business (Office of Government Commerce, 2011a). Hence, for a systems engineer, making the correct decision is crucial in order to decide which incident has to be dealt with first in order to keep the impact on the respective business as minimal as possible. Therefore, we applied AHP methodology to support the prioritization process regarding incident management based on well-defined criteria. Furthermore, we developed a software tool relying on innovative cloud technology which delivers decision-support based on the AHP prioritization algorithm. There is a lot of ITIL-compliant software commercially available that is used by IT service providers to support their business (ITIL, 2014). However, to the best of our knowledge, no ITIL-compliant software exists that relies on a robust mathematical foundation for prioritizing IT-related incidents.

3. Methodology

The Incident Prioritizer (IP) software calculates priorities for incidents on the basis of an AHP decision model. This model has the goal of finding the incident with the greatest business impact. Thus, the decision model considers incidents as its alternatives. It is important to understand that the decision model used in the IP is not static. It varies depending on the management's preferences regarding criteria as well as on the current open incidents of an IT service provider organization.

The criteria that can be used in the decision model have been derived from expert knowledge. The expert is working at a large IT service provider organization in Austria that serves customers all over the globe. Based on interviews with this expert the following four criteria were developed:

- **NoOfCustAffected** – The number of customers affected by an incident.
- **IsImpCustAffected** – Is a customer marked as important affected by an incident?
- **Penalty** – The total penalty caused by an incident.
- **RemainingTime** – The time remaining until a service level target is violated.

The Incident Prioritizer software allows the management of the IT service provider to choose from the aforementioned four criteria the ones they think influence the business impact of incidents most. The chosen criteria are then considered in the AHP decision model for the calculation of priorities. The judgments of the criteria with respect to the goal have to be carried out on an individual basis by the management of the IT service provider that is using this decision support system. This step is carried out via the so called frontend module of the IP software (see Section 5.1). The judgments made there will reflect the management's understanding of what contributes most to the business impact caused by incidents. The judgments of the alternatives (incidents) with respect to the criteria are carried out automatically by the IP software based on so-called criteria data. The term criteria data refers to numerical data that is processed by the IP software to obtain judgments for alternatives with respect to upper level criteria.

4. Decision model analysis

As already mentioned in the previous section, the IP software uses an AHP decision model to calculate priorities for incidents. Figure 1 shows a representative example of an AHP decision model that can be successfully processed by the IP software to calculate priorities.

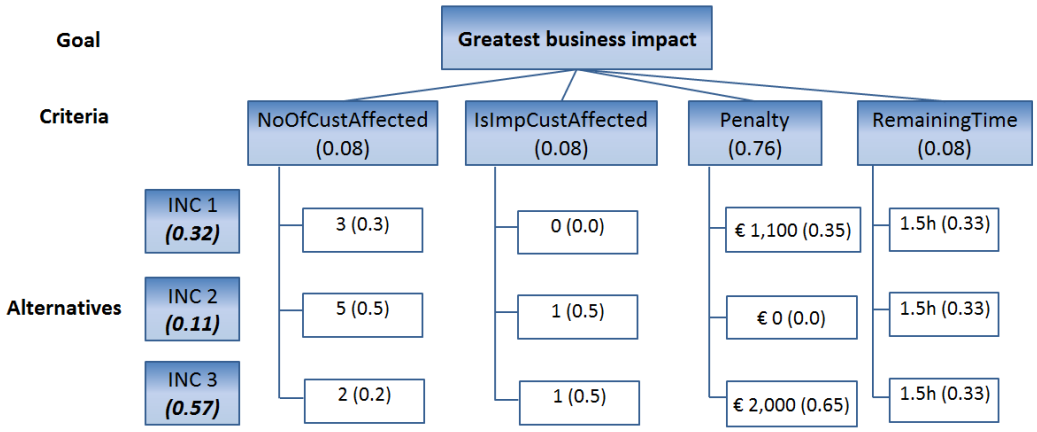

Figure 1. Representative example of an Incident Prioritizer compliant AHP decision model

The decision model shown in Figure 1 with its criteria, their weightings and the alternatives was defined by the authors of this paper. The weightings, the alternatives and their related data are all fictional and arbitrary. Thus, the example decision model does not claim to be an accurate model to provide a real world IT service provider with optimal incident priorities, but to be a valid model that helps explain how the Incident Prioritizer software functions. However, as already mentioned, in a real world application of the Incident Prioritizer the management of an IT service provider organization will define the criteria, their weightings and the alternatives according to their business goals. The example decision model in Figure 1 assumes that a service provider's management decided to consider all of the four well-defined criteria in the prioritization process. The criteria weightings that the management made in this example are shown in brackets underneath the names of the criteria (e.g. 0.76 is the weight for the criterion "Penalty"). The criteria weightings were obtained using Saaty's concept of pairwise comparison. The IP software provides a graphical user interface called "Frontend module" that enables its users to pairwise compare criteria and thus enables them to define criteria weightings (see Section 5.1). In this example, the management puts the highest weight on the criterion "Penalty" when it comes to assess the business impact of incidents. Figure 1 shows three incidents as alternatives that have to be prioritized (i.e. "INC 1", "INC 2" and "INC 3"). The white boxes that are connected to the upper level criteria represent the so-called criteria data of the incidents (e.g. 3, 5, and 2 for the criterion "NoOfCustAffected"). The content of the top left white box in Figure 1 has the following meaning: incident 1 ("INC 1") affects 3 customers. As compared to the other two incidents and how many customers they affect, the priority of incident 1 with respect to the criterion "NoOfCustAffected" is 0.3. The final priorities of incidents are also shown in brackets underneath the name of the incidents (e.g. 0.57 is the final priority for incident "INC 3"). All calculations of priorities are carried out by the Incident Prioritizer software.

As already mentioned in Section 0, the judgments of the alternatives with respect to the criteria do not come from human beings. Instead, the judgments are based on the aforementioned criteria data, and are obtained by putting the criteria data of incidents in relation to each other. For instance, let's compare INC1 to INC2 with respect to the criterion "NoOfCustAffected" from Figure 1. The criteria data for incident 1 with respect

to "NoOfCustAffected" is 3. The criteria data for incident 2 is 5. The judgment is obtained by dividing the criteria data of incident 1 by criteria data of incident 2 (3/5). This step is carried out by the Incident Prioritizer software. The following section will provide details on the technical implementation of the Incident Prioritizer software.

5. Implementation details

The Incident Prioritizer is part of a software architecture that involves state-of-the-art cloud computing technology. Figure 2 illustrates the complete software architecture of which the Incident Prioritizer is a part.

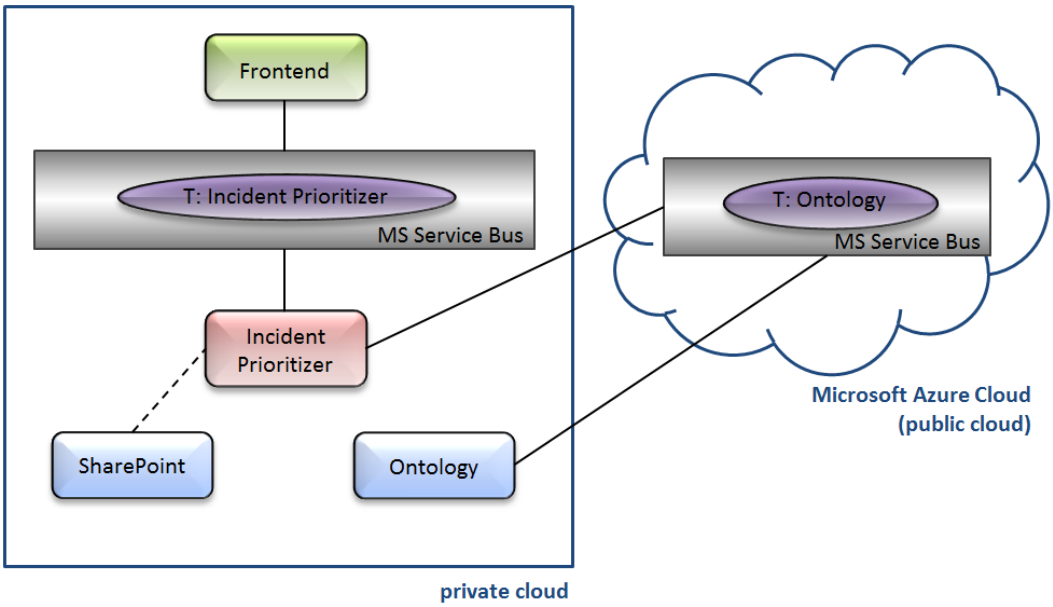

Figure 2. Overall architecture of the Incident Prioritizer software

Figure 2 exhibits a hybrid cloud architecture that consists of a private and a public cloud. Both clouds incorporate a Microsoft service bus component. The private cloud additionally contains the modules "Frontend", "Incident Prioritizer", "SharePoint" and "Ontology". The modules "Frontend" and "Incident Prioritizer" are connected to the service bus in the private cloud. The module "Incident Prioritizer" is also connected to the service bus in the public cloud as is the "Ontology" module.

The service bus exhibits a publish/subscribe messaging behavior. It orchestrates the communication between modules that are connected to the service bus. The connected modules communicate via so called "topics". In Figure 2 the topics are indicated with the "T" letter. An extract of the Incident Prioritizer data flow illustrated in Figure 3 will help explain how the service bus works in the context of the Incident Prioritizer.

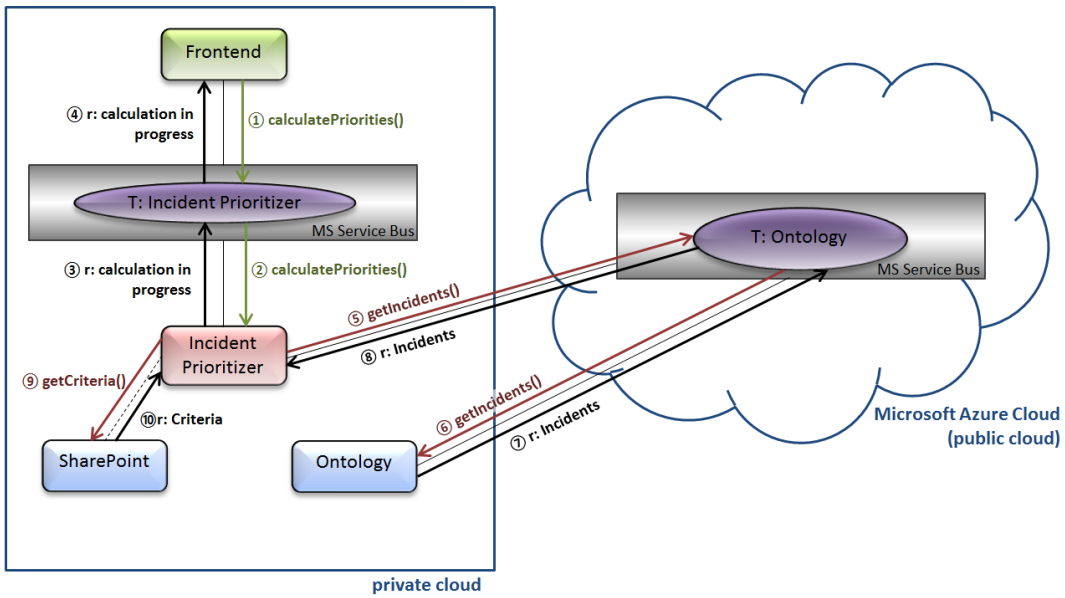

Figure 3. Extract of the data flow within the IP software architecture

In step 1 of Figure 3 the frontend publishes a calculatePriorities() message on the Incident Prioritizer topic of the service bus located in the private cloud. In step 2, the IP module receives this message due to its subscription on the Incident Prioritizer topic. Upon receiving the calculatePriorities() message, two actions are triggered within the IP module. The first action is a response message (step 3) that is published back to the topic to inform the frontend (step 4) that the calculation is in progress. The second action spawns a process thread within the IP module that carries out all the further steps of the calculation. In Figure 3 those calculation steps start with step 5 by querying the ontology topic in the public cloud for incidents. Since the ontology module located in the private cloud is subscribed to the ontology topic in the public cloud, it receives the incident query in step 6. In step 7 it responds by publishing the requested list of incidents back on the ontology topic where the IP module is also subscribed to. In step 8 the IP module receives the list of incidents and continues its calculation process by querying the SharePoint module directly via its interface for criteria in step 9. In step 10 the SharePoint module responds back to the IP module with a list of criteria that the IP module will consider later in its calculation process. The results of the priority calculations of the IP have to be requested by the frontend module. This step is not shown in Figure 3.

This centralized communication via a service bus has certain benefits such as a unified communication interface for each module connected to the service bus. Figure 2 and Figure 3 exhibit a service bus in the public cloud. It shows that both the IP and the ontology module are located in a private cloud and are interacting with the ontology topic in the public cloud. While the given problem of prioritizing incidents does not necessarily demand such a sophisticated hybrid cloud architecture it could be valuable for other business scenarios, e.g. collaboration between two companies, or outsourcing of non-critical business information into the cloud for cost cutting reasons.

The architecture shown in Figure 2 and Figure 3 features the following technologies for its components:

- Service bus (public cloud) – Windows Azure Cloud[2]
- Service bus (private cloud) – Windows Azure Pack for Windows Server
- Incident Prioritizer module – Windows Service
- SharePoint module – Microsoft SharePoint 2013
- Frontend module – Windows 8.1 App
- Ontology module – Java-based web service

While the IP module in this architecture interacts with the ontology via the public cloud-based service bus and SharePoint, the description of those two modules is beyond the scope of this paper. Upcoming publications will deal with these topics in detail as well as a description of the frontend module. Details on the service bus in the private and public cloud will also be published separately.

After describing the overall architecture with its data flow mainly coordinated via the service bus, the following sections will outline the modules of the architecture shown in Figure 2 in more detail. Emphasis will be placed on the Incident Prioritizer module since it is the software that incorporates the AHP method that is the focus of this paper.

5.1 Frontend Module

The frontend module provides a graphical user interface for using the functionality of the Incident Prioritizer. Technically, it is a Windows 8.1 application that can be used to trigger the calculation process of the Incident Prioritizer and to display the resulting incidents and their priorities. Figure 4 shows a screenshot of the view on the incidents.

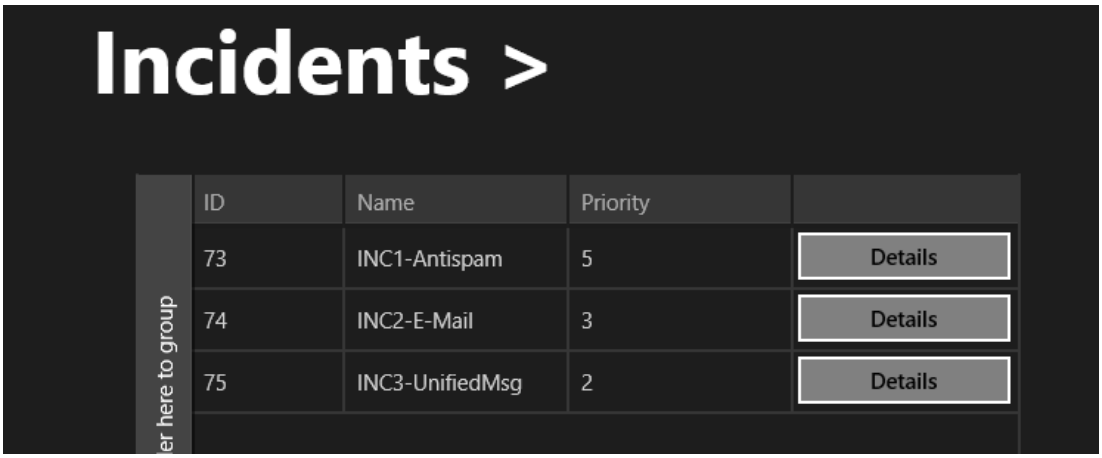

Figure 4. Frontend – Incident list

[2] Microsoft, SharePoint, Windows, Windows Server, and Windows Azure are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries.

Figure 4 illustrates a list of prioritized incidents. It shows that the incident with the ID 73 has the name "INC1-Antispam" and a priority of 5. The priority scheme used here ranges from 1 being the highest and 5 being the lowest priority. This information screen is highly relevant for the service engineer's decision about which incident needs to be resolved first. The frontend module also allows forecasting priorities of incidents up to 18 hours. The forecast screen is shown in Figure 5.

Figure 5. Frontend – Forecast priorities

The frontend module also provides the user interface to customize calculation parameters such as criteria and their priorities that should be taken into consideration in the calculation process. This interface is used by the management of the IT service provider to express their understanding of the business impact of incidents. Figure 6 shows a screenshot of this interface.

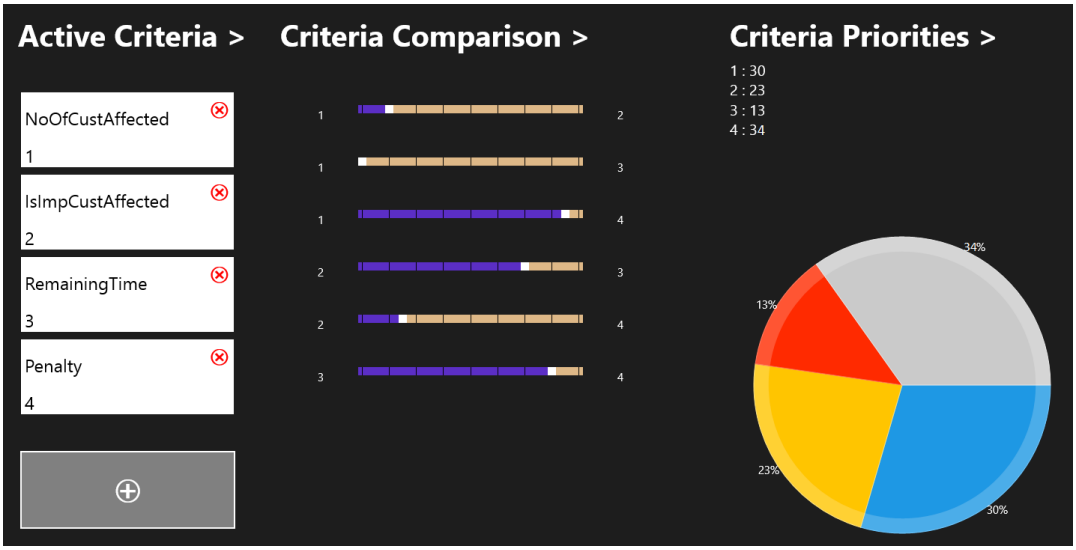

Figure 6. Frontend – Criteria customization

The screen in Figure 6 shows that four criteria are active and are to be considered in the prioritization process. The middle part of the screen shows the pairwise comparisons of the active criteria. The pairwise comparisons are based on Saaty's scale of absolute numbers (Saaty, 2008), and result in criteria priorities. These criteria priorities are visualized in a pie chart in the right part of the screen. The calculation parameters defined here translate into the criteria level of the AHP decision model (as shown in Figure 1) that is used later for the calculation of incident priorities. The frontend module communicates with the Incident Prioritizer module via the service bus component to trigger the calculation process, fetch data like the results of the calculation and change calculation parameters such as criteria.

5.2 Incident prioritizer module

This section describes the Incident Prioritizer module that was developed for this study with the goal of prioritizing incidents according to the severity of their business impact by using the AHP method.

The IP module was developed in the programming language C# and its internal structure follows the idea of separation of duties. Thus, the overarching task of applying the AHP method for prioritizing incidents based on calculations that require data from different sources was broken down into four distinct components. Figure 7 shows these components.

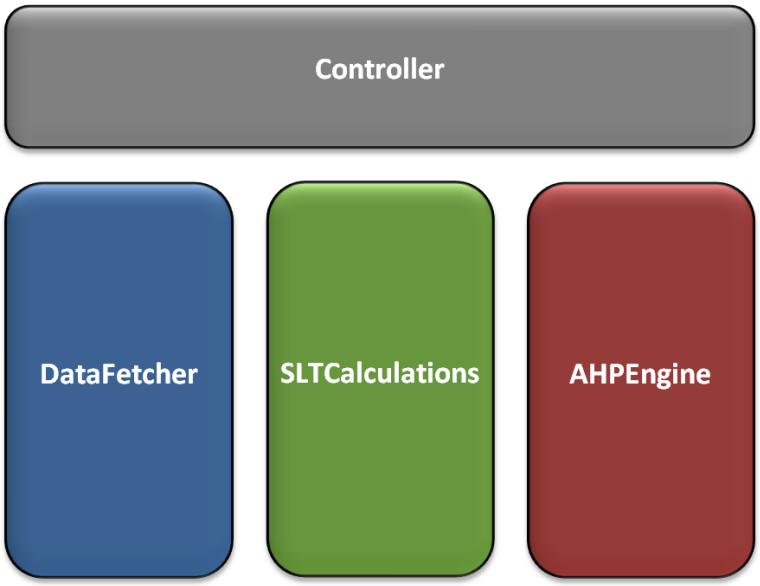

Figure 7. Components of the Incident Prioritizer

In Figure 7, the Controller component defines the messages that can be used to control the IP module from outside by the frontend module (e.g. to calculate priorities). Also it coordinates the data flow between the other three components. The DataFetcher component is responsible for the communication between the IP module and the other modules that act as data sources for the IP, i.e. SharePoint and ontology. Despite the name DataFetcher, it does not exclusively fetch data from the named sources but also pushes back, for example, result data and priorities to the SharePoint module. Furthermore, it acts as a provider for criteria data for the criteria "NoOfCustAffected" and "IsImpCustAffected" that is fed into the AHPEngine component. The SLTCalculations component implements algorithms necessary to calculate criteria data based on service level targets information. To accomplish this, the SLTCalculations component communicates with the DataFetcher component to get service level target data from the ontology. Like the DataFetcher, the SLTCalculations component acts as a provider for criteria data for the criteria "RemainingTime" and "Penalty". The AHPEngine component is responsible for calculating priorities based on data about incidents, criteria with their priorities and criteria data, and it uses the AHP method to obtain priorities. Therefore, all the data available for an incident and its respective service are mapped into an Incident Prioritizer compliant AHP decision model that is described in Section 4 before carrying out the calculation. Since the focus of this paper is the application of the AHP method, the implementation of the AHPEngine component of the Incident Prioritizer module will now be explained in more detail.

A part of the AHPEngine component is based on an open-source implementation of the AHP method. The Open Decision Maker (ODM) is an implementation of AHP that also includes a graphical user interface to design and calculate an AHP decision model. It was written in the programming language Java. The authors of ODM are Bender, Blocherer, Rossmehl, Rotter (ODM, 2010). For the AHPEngine component parts from the Java source code of ODM 1.0.1 were taken and translated into the programming language C# since this is the language in which the Incident Prioritizer module was developed. The

Free Edition of the software tool "Java to C# Converter" was used to support this translation process. The ODM implementation of AHP imposes limits regarding the maximal possible number of criteria that can be used in the decision model. According to the documentation included in the source code of ODM 1.0.1, the maximum possible number of criteria is 24. The AHPEngine component is represented by the C# class AHPEngine. This class offers the single point of access to the AHP functionality provided by the ODM AHP implementation within the Incident Prioritizer module.

The AHPEngine class imposes the additional limit that only one level of criteria can be used to determine priorities for incidents. Thus, sub-criteria that are normally possible in the AHP method are not possible in the application of AHP in the Incident Prioritizer.
The AHPEngine class provides all methods necessary to set up the required AHP decision model for the task of incident prioritization. These methods are:

(1) addCriterion() – to add a criterion to the decision model.
(2) addAlternative() – to add alternatives to the AHP decision model. In the case of the Incident Prioritizer implementation the alternatives are incidents.
(3) setComparisons() – to add the pairwise comparisons for criteria that are necessary to calculate priorities of the criteria.
(4) setAltComparisonsBasedOnDirectData() – to add the criteria data (e.g. number of customers).
(5) calculatePriorities() – to determine the final priorities of the alternatives (incidents) from the set up AHP decision model.

Figure 8 shows what these methods are used for in the context of the AHP decision model introduced in Section 4.

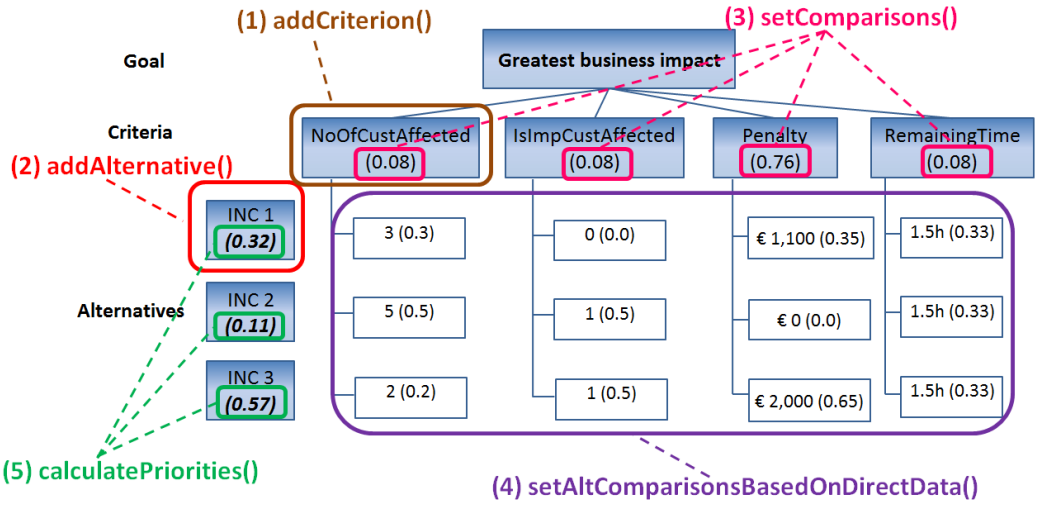

Figure 8. AHPEngine methods to set up an AHP decision model

Figure 8 shows the effect of the AHPEngine methods that are used to set up an AHP decision model within the Incident Prioritizer software module. The (1) addCriterion() method is called for each criterion that should be considered in the prioritization process. The criteria to be used here are restricted to the four well-defined criteria that were

introduced in Section 0. The (2) addAlternative() method is called for each alternative to be considered. In the context of the Incident Prioritizer software the alternatives to be considered are incidents. The example in Figure 8 shows three incidents to be considered for prioritization that are abbreviated as "INC 1" to "INC 3". The (3) setComparisons() method is called once in each prioritization process. As already explained in Section 4, pairwise comparisons of the criteria are carried out by the management of an IT service provider. By calling (3) setComparisons() the results of pairwise comparing the criteria are transformed into the criteria weightings (e.g. weight 0.08 for criterion NoOfCustAffected). The (4) setAltComparisonsBasedOnDirectData() is also called only once in each prioritization process. Similar to the setComparisons() method the setAltComparisonsBasedOnDirect-Data() method transforms the criteria data (e.g. 3, 5, 2 for NoOfCustAffected) after pairwise comparing them into priorities with respect to the upper level criterion (e.g. priorities 0.3, 0.5, 0.2 for upper level criterion NoOfCustAffected). This process of pairwise comparing the criteria data was already described in Section 4 of this paper. Once the decision model is fully built with criteria, alternatives and criteria data, the (5) calculatePriorities() method is called at last to obtain the final priorities of incidents (e.g. incident "INC 1" has the final priority 0.32).

The final priorities obtained from the AHPEngine are later translated into absolute numbers ranging from 1 (highest priority) to 5 (lowest priority). The translation to absolute numbers is performed in order to make the priority information easier to understand for the people that are using the Incident Prioritizer decision support system. Furthermore, integers ranging from 1 to 5 perfectly match priority codes of the incident priority matrix as described elsewhere (Office of Government Commerce, 2011a). The prioritized incidents can finally be viewed in the frontend module that was already introduced in Figure 4.

5.3 SharePoint Module

In the context of the Incident Prioritizer, this module is used like a database. It stores the data for criteria and criteria weightings that are considered by the IP module in its AHP-based calculation of priorities. Also the SharePoint module stores the results of the priority calculation process carried out by the IP module. Figure 9 shows a screenshot of the SharePoint List "Criterion". It shows the four criteria that can be taken into consideration by the Incident Prioritizer.

Incident Prioritizer EDIT LINKS Search this site

Criterion

new item or edit this list

All Items ... Find an item

ID	Name	CodeName	Description	IsActive	InterfaceName
1	Number Of Customers Affected	NoOfCustAffected	This criterion refers to the number of customers that are affected by a incident.	Yes	DataFetcher
2	Are Important Customers Affected?	IsImpCustAffected	This criterion assigns incidents a higher priority that affect customers considered to be important.	Yes	DataFetcher
3	Remaining Time Until SLO Violation	RemainingTime	This criterion refers to the calendar hours that are left until any of the SLOs of a service is violated.	Yes	SLTCalculations
4	Penalty	Penalty	This criterion refers to the penalty a incident has already caused. This is a money value.	Yes	SLTCalculations

Figure 9. SharePoint List "Criterion"

5.4 Ontology module

The ontology module provides the Incident Prioritizer with so-called incident-related data. The term incident-related data is used within the context of the Incident Prioritizer and includes for example data about customers, contracts, services and service level targets that are affected by an incident. After querying this incident-related data from the ontology the Incident Prioritizer module processes it internally into criteria data. As already explained in Section 4, criteria data is crucial to obtain judgments for the alternatives with respect to the criteria.

The ontology module hosts an ITIL-compliant service catalog based on semantic technologies (Office of Government Commerce, 2011b). In the discipline of IT service management and it's most popular framework ITIL, the service catalog is a central component that brings together the different perspectives on the IT service that is provided by an IT service provider. The perspectives include among others: customers, customer contracts with agreed service level targets, the IT services offered to potential customers, the IT services consumed by actual customers, the technical dependencies of the IT services offered (e.g. hardware, software), etc. The concept and the implementation of this module were introduced by Wagner (2012). Figure 10 shows an extract of the service catalog model that is implemented by the ontology module.

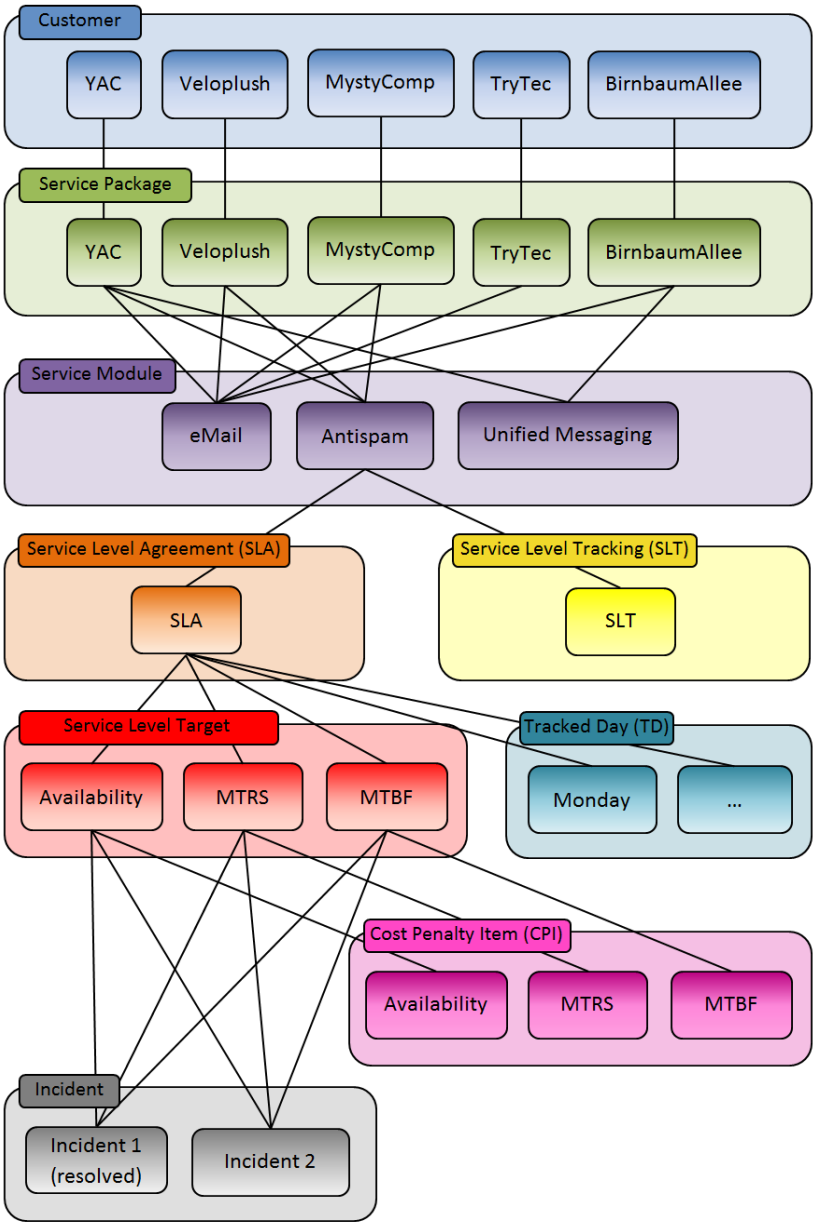

Figure 10. Extract of service catalog of ontology module

The service catalog shown in Figure 10 consists of entities of certain types. The entity "YAC" for example is of type "Customer" and the entity "Antispam" is of type "Service Module". Each entity type has a defined set of data attributes. Entities of type "Customer" for example have the following set of data attributes: name, address, ZIP code, and country. The service catalog also models relationships between these entities. The entity "Antispam" for example is related to the entity "SLA".

By establishing relationships between the entities in the service catalog valuable business information is generated that can be further processed by other information systems such as the Incident Prioritizer module. In the case of the Incident Prioritizer, the ontology

offers the highly valuable incident-related data that is necessary for generating criteria data. As described in Section 4, criteria data is crucial for obtaining the judgments for alternatives (incidents) with respect to the criteria.

Technically, the Incident Prioritizer module queries the ontology for this information by using the SPARQL language that is designed to retrieve information from ontologies.

6. Limitations

So far, the developed decision support system has been tested with only three synthetic incidents. However, in a real business scenario the system needs to be able to handle dozens or even hundreds of incidents which might lead to a non-deterministic behavior of the software module. Another limitation concerns the consistency of judgments made within the IP software. By design of the IP judgments are made on the criterion level by human beings as well as by the software itself on the alternatives level. Hitherto, the IP does not check the consistency of the judgments on any of those two levels.

7. Conclusions

In this study a decision support system called Incident Prioritizer was developed. It implements the AHP method for the purpose of prioritization. In particular, the Incident Prioritizer is an ITIL-compliant software tool that can be used by IT service providers to prioritize incidents based on the severity of their business impact.

The Incident Prioritizer was presented as part of a software architecture that features state-of-the-art cloud computing technology. The Incident Prioritizer was developed in the programming language C# and requires three other software modules (frontend, ontology, SharePoint) for proper functioning. This paper introduced the AHP decision model that is used by the Incident Prioritizer software to calculate priorities for incidents. Commonly, Incident Prioritizer compliant decision models are restricted to only one level of criteria. Furthermore, the judgments for the criteria have to be carried out by the management of the IT service provider whereas the judgments for the alternatives with respect to the criteria are carried out programmatically by the Incident Prioritizer on the basis of criteria data.

Future research should test the developed decision support system in a real business scenario to find out whether the software provides meaningful results when facing the challenge of prioritizing dozens of incidents. Another suggestion for further research is to check for the consistency of judgments that are considered in the prioritization process.

REFERENCES

Beims, M. (2010). *IT-Service management in der praxis mit ITIL 3*. Munich: Hanser Verlag.

ITIL. (2014). *Endorsed software tools*, http://www.itil-officialsite.com/SoftwareScheme/EndorsedSoftwareTools/EndorsedSoftwareTools.aspx (Accessed March 31, 2014)

Office of Government Commerce (OGC). (2011a). *ITIL® Service Design.*

Office of Government Commerce (OGC). (2011b). *ITIL® Service Operation.*

OGC. (2007). *ITILv3 - Service operation.* Norwich: The Stationary Office.

ODM. (2010). *Open decision maker - User manual* [PDF-File], http://sourceforge.net/projects/opendecisionmak/files/latest/download?source=files (Accessed May 20, 2014)

Orta, E., Ruiz, M., Hurtado, N. and Gawn, D. (2014). Decision-making in IT service management: a simulation based approach. *Decision Support Systems, 66*, 36-51.

Power, D.J. (2002). *Decision support systems: Concepts and resources for managers.* Westport, CT: Quorum Books.

Repschlaeger, R., Proehl, R. and Zarnekow, R. (2014). Cloud service management decision support: An application of AHP for provider selection of a cloud-based IT service management system. *Intelligent Decision Technologies, 8*, 95–110.

Saaty, T.L. (2008). Decision making with the analytic hierarchy process, *International Journal of Services Sciences, 1(1)*, 83–98.

Wagner, G. (2012). *Semantik ITSM* (Master thesis). Graz: FH JOANNEUM University of Applied Sciences.

Wan, J. and Wan, X. (2012). Case study on M company best practice with global IT management. *Technology and Investment, 3*, 143-148.

Wan, J., Zhang, H. and Wan, D. (2011). Evaluation on Information Technology Service Management Process with AHP. *Technology and Investment, 2*, 38-46.