Applied Mathematics & Information Sciences
An International Journal

Stability Analysis of Wireless Sensor Network Service via Data Stream Methods

Ruo Hu

Guangdong Polytechnic Normal University, Guangzhou, 510665 P. R. China
E-mail: hu68@163.com

Received: 11 Feb. 2012; Revised 16 May 2012; Accepted 01 Jul. 2012

Abstract: Providing stable compositions of Wireless Sensor Network Service (WSNS) is a challenging issue since the data stream architect often has only a limited control over the stability of the composed services. The architect can instead achieve stability by properly planning the data stream architecture. In this paper, we propose a formal approach which allows a data stream architect to perform stability analysis of WSNS-based security service. The approach exploits the concept of stability methods to evaluate the stability function of a wide class of data streams. The effect of the approach and the operation of the tool are demonstrated with respect to a case study of a business security infrastructure realized by simple security services.

Keywords: Security, Wireless Sensor Network Service, Stability Analysis, Data stream methods

1 Introduction

Wireless Sensor Network Service (WSNS) can be combined in complex composite services achieving new functionalities [1]. Composition may aggregate security services developed and exposed within a certain organization. More interestingly, the composed Wireless Sensor Network Service can be the result of an edit of services exposed by different organizations. Benefits of composition have been long discussed in the past few years highlighting and demonstrating the advantages coming from the achievement of new functionality by composing autonomous services [2]. Nevertheless it is widely accepted that web applications are easy to fail as confirmed by a U.S. Government study [3] and for Wireless Sensor Network Service the situation is also worst since a number of application layers are built on top of classical web servers. As pointed out in [4], failures are inevitable in the modern Internet-Connected environments and, when dealing with composite services, assuming the failure of any individual Web Service will cause the failure of the composite service, even if all the other Wireless Sensor Network Service are stable, one unstable Web Service could decrease the overall stability to a very low level. This evidence related to the stability of the composite Web Service rises up a doubt with respect to the actual adoption of this distributed model of developing complex services [5]. Since the stability engineer designing the data stream has no chance to modify the simple services at all, especially while dealing with services composed across organization boundaries, the only way to ensure the stability of the composite service is increasing the stability of the data stream by appropriately planning its architecture, i.e. properly adopting diversity and redundancy. This requires the development of appropriate methodologies for a quick and early evaluation of the composite service stability and the development of tools which can be easily adopted to compare multiple architectural choices for the edit of a service. In this paper, we propose a formal approach that allows a data stream architect to perform stability analysis of a Wireless Sensor Network Service-based service. The approach exploits the concept of stability methods to derive an aggregate stability function and it is suited for a wide class of data stream processes. The approach is implemented in a tool-Active BPEL（Business Process Execution Language）. The tool allows the system architect to evaluate the impact—in terms of stability—of possible data stream alternatives, as early as in the first steps of the design. The effectiveness of the approach and the operation of the tool are demonstrated with respect to a case study of a business security infrastructure realized by orchestrating simple security services. The rest of the paper is organized as follows. Section 2 provides an overview of the related work. Section 3 presents the concept of data stream methods, while in Section 4 stability methods are derived and their stability formulas evaluated. Section 5 discusses the assumptions and limitations of the model. Section 6 presents a typical case study can be applied in the field of stable data stream development. In Section 7 it is described the implementation of the component for Active BPEL and its operation is demonstrated with respect to the case study at hand. Finally, Section 8 concludes the paper with final remarks.

2 Related Work

Wireless Sensor Network Service based systems are typically composed by orchestrating a number of simpler services (generally Wireless Sensor Network Service themselves) in a common data stream. In such a case it is widely accepted that the stability function of the data stream must be derived based on the stability functions of individual tasks in the data stream. A mature work in this field is [4], where the authors propose a set of data stream methods with related stability expressions. A data stream engine named Web-star, which allows combining such methods to build a more complex data stream, is also presented. Based on the stability expression of the elementary methods, Web-star permits to derive the aggregate stability expression of the composed data stream. The main limitation of this approach lies in the possibility of getting the stability expression only for those data streams that can be obtained by composing the methods described in [5]. To overcome such limitation, we start from results presented in [3] where the authors present, by extending results reported in [2], a set of 43 data stream methods whose combinations can provide pretty every data stream. Starting from this set of methods, we identify the combinations of them that are meaningful from a stability point of view and derive for them a stability expression. We refer such combinations as “stability methods”. Since virtually any data stream can be obtained by combining the data stream methods, “stability methods” can also be applied to obtain the stability formula for any data stream. We demonstrate our approach to derive new stability methods from the remaining methods defined in [5]. By doing so we verify that, not only all the methods defined in [4] are also obtained as stability methods, but new methods, not considered in [4], such as the “Multi-Merge Parallel”, are also identified and considered for stability evaluation. Since our approach can be applied to retrieve the stability expression of virtually any data stream, it can be applied to already existing data stream designing tools, instead of needing the design of new ones, as it was for Web-star. This is verified by applying the concept of stability methods to a popular commercial data stream designing tool, namely Active BPEL Designer, and enabling an early evaluation of stability formulas for designed data streams. Even more interestingly, the proposed component can be easy adapted to any WS-BPEL compliant designer. Finally it is worth noting that the estimation of such formulas is not intended at exactly measuring the stability of a composed service, but at allowing a hypothesis analysis of alternative architectural solutions at design time. This means that the simplicity of stability expressions should be preferred to their precision.

3 Data stream Methods

E. Gamma et al. defined a method as “The abstraction from a concrete form which keeps recurring in specific non arbitrary contexts”. When dealing with Wireless Sensor Network Service composition it is worth considering data stream methods that are defined as “An abstract description of a recurrent class of interactions based on activation dependencies”. Data stream methods can be considered from multiple perspectives, namely a control-flow perspective, a data perspective, a resource perspective, an operational perspective. In particular control-flow perspective refers the execution order of a set of activities. With respect to the control-flow perspective of data stream, M. da S. identifies a set of twenty basic control-flow methods (in the following referred as methods), which can be combined to generate virtually any control flow. While analyzing the methods provided above, some observations are due: 1) Since we are only interested in the stability of the data stream from an architectural point of view, not all the methods are relevant for our purposes. As an example, the method Cancel Case relates to the data stream management system and is therefore not relevant for the composition process. 2) From a stability point of view some methods are equivalent. As an example the Multiple Instances method provides the same stability of a Parallel Split or of a Multiple Choice depending on the necessity of completing or not all the activated instances. 3) Combinations of methods are often needed—in order to address stability—instead of individual methods. This is the case of the Parallel Split, for which deriving stability requires knowing if the following task is Synchronization or a Multi-Merge.

4) Finally, not all method combinations yield valid data streams, as one example the sequence of a XOR-Split and an AND-Join is not allowed since it refers to a scenario, where only one in a set of tasks is activated but the end of all of them is waited before the data stream can terminate. In the next sections, we first describe an algorithm which derives the aggregate stability function through a data stream graph reduction, then we discuss the derived stability methods and their stability formulas, finally we present an example showing how the algorithm works.

4 Stability Analysis

We define the concept of stability method as: "An elementary combination of methods which itself behaves as a method from a stability perspective". The previous definition yields that 1) for a stability method, a stability formula can be defined starting from the stability formula of each activity in the method, 2) for any subset of methods in the stability method, a related stability formula cannot be defined. As an example the sequence of an AND-SPLIT method and a MULTI-MERGE JOIN method matches an m-out-of-n stability structure so it can be uniquely characterized from a stability point of view. On the other hand it is not possible to characterize the stability of neither the AND-SPLIT method nor the MULTI-MERGE JOIN method if they are considered separately. In the following sections, first is defined a reduction algorithm exploiting the concept of stability methods to characterize the stability of a data stream, then stability methods are identified and their stability formulas are obtained.

4.1 Reduction Algorithm

When dealing with a data stream, we are assuming that Wireless Sensor Network Service is composed in an edit. We assume a data stream described as

$$W = (t, a, fr, fp, fc) \qquad (1)$$

Where: t is a set of tasks (each represented by a circle); a is a set of transitions (each represented by an arrow); fr is a function which associates to every task t_i in t its stability function; fp is a function which associates to every transition aij (connecting the task i to the task j) a probability p_{ij}, representing the probability that once task t_i terminates task t_j is activated. In other words p_{ij} represents the probability of activation of the transition a_{ij}. Every time the task t_i is unambiguously identified, the index "i" will be omitted and p_{ij} substituted with p_j; and fc is a function which for every task t_i in t associates a value c_i representing the probability that a failure of task t_i does not lead to a failure of the data stream. Hence c_i represents a coverage factor, and can be expressed as:

$$c_i = \sum_{g \in G} \phi(g) P(g) \qquad (2)$$

Where:

g is a failure mode for the task i;

G is the fault dictionary for the task i;

ϕ(g) = 1 if the failure g can be tolerated, 0 otherwise;

P(g) is the occurrence probability of the failure g;

This implies that the stability for the single task is increased by a factor representing the probability that the component will fail without leading to a data stream failure, that is:

$$R_i = R_i' + (1 - R_i')c_i \qquad (3)$$

Where R_i' represents the stability of the task t_i and represents the stability of the task t_i as perceived by the data stream engine. The latter equals the former when c is zero, i.e. the data stream cannot tolerate a fault in one of the edited services. If c equals 1 the formula returns 1 meaning that the component is optional from a stability point of view. In the next two sub-sections we will always use the term stability with reference to the meaning it assumes in (3). A start task and an end task must be identified into the set of the tasks. The start task does not have any incoming transition and represents the invocation of the edited service by an external client. The end task does not have any outgoing transition and represents the end of the edit. Once the graph representing the Wireless Sensor Network Service edit is defined, the reduction algorithm is performed by going backward through the graph (from the end task to the start one) and each time an individual stability method is found its component tasks are collapsed in a single task whose stability is defined by the method stability formula. The process is than iterated until the whole data stream is collapsed in a single task whose stability depends on the stability of the individual tasks, the probabilities p_{ij} and the coverage factors c_i .

The authors described the identified stability methods in a previous work, in Table 1 are reported the formulas of these methods.

5 Assumptions and Limits of the Model

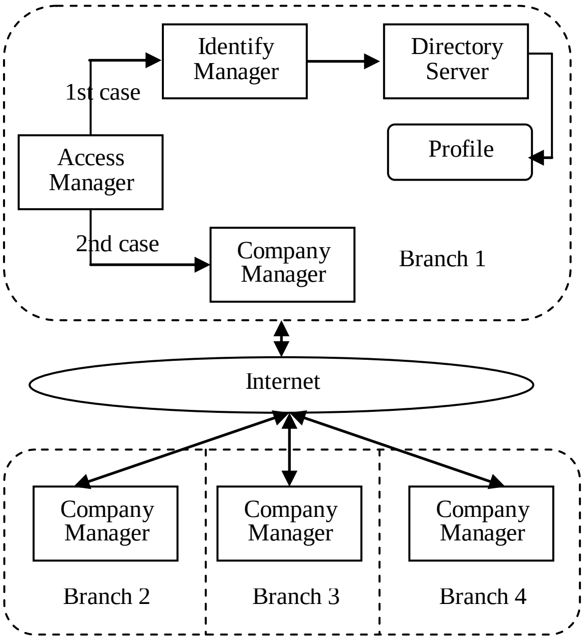

Figure 1. four branches in a domain.

The main hypothesis underling the analysis proposed in the previous section, and of course the proposed approach, is the independence of events A_i = 'time to first failure of activity i' $\in$ t and A_j= 'time to first failure of activity j' $\in$ t , for each I $\neq$ j. This means, for example, that if two services are offered by the same provider it is assumed that they are deployed on physically independent servers. A further simplification in this approach lies in the absence from the model of the communication channel stability. Actually the communication channel may itself introduce faults, as an example by dropping packets, or modifying them or just delaying their delivery beyond timeout expiration. Anyway such a kind of behavior can be embedded into the model of the single service. Finally it is worth noting that the obtained model provides the stability of the services edit without considering the stability of the service that performs the edit. As an example let us consider a service which by means of an edit engine (e.g. BPEL) coordinate the invocation of other services by following a predefined data stream. In this case the stability of the edit service, of the server hosting such a service and of the edit engine, should be modeled and in case of hypothesis of independence it should be multiplied by the stability of the entire data stream.

Table 1. Stability method expressions.

Stability Method Name	Stability Method Expressi
Sequence	$R = R_A x R_B$
Synchronizing Parallel	$R = R_A R_B \sum_{i_i=0}^{2} \cdots \sum_{i_N=0}^{2} u\left(\sum_{j=1}^{n} \partial(i_j - 1) - k\right) \times \times \prod_{j=1}^{n} \left[\partial(i_j - 1) R_j p_j + \partial(i_j)(1 - R_j) p_j + \partial(i_j - 2)(1 - p_j)\right]$
	$\sum_{j=1}^{n} \partial(i_j - 1) > k \Leftrightarrow \sum_{j=1}^{n} \partial(i_j - 1) - k > 0$
Multi-merge Parallel	$R = R_A \sum_{i_1}^{2} \cdots \sum_{i_n=0}^{2} u\left(\sum_{j=1}^{n} \partial(i_j - 1) - k\right) \times \times \prod_{j=1}^{n} \left[\partial(i_j - 1) R_j p_j R_B + \partial(i_j)(1 - R_j) p_j + \partial(i_j - 2)(1 - p_j)\right]$

6 Case Study

This section considers a realistic case study to show how the proposed approach can be applied in the field of stable data stream development. The case study considers a company with four branches each with its IT department. The four branches are federated in a trusted domain. So that a user which logs at one of the federated entities, in order to access a service, obtains a SAML token which can be used at a later time to log in at any of the other federated entities without the need of being authenticated again. In the presented scenario the user, holding the SAML (Security Assertion Markup Language) token, sends a request to access a security service provided by the company (Figure 1). The request is intercepted by the Access Manager of one federated entity which picks the SAML token up and tries to validate it. Two alternatives are possible:

The SAML token was actually released by that entity (branch 1 of the company): in this case, the Access Manager requests the Identity Manager to retrieve the appropriate authorization profile for the user holding the token. The Identity Manager will do it by means of the Directory Server which provides an abstraction of the data repositories in the company.

The SAML token is not recognized as a valid token by the branch 1 of the company: the Access

Manager charges the Federation Manager in its domain with managing the SAML token. The Federation Manager will ask other Federation Managers in the same trusted domain to check for the SAML token. Federation Manager will provide its Access Manager with a copy of the token. These in turn will repeat the same steps of the validation procedure as operated by the Access Manager in the branch 1. Finally the required service will be accessed with the authorization profile provided by the federated entity which actually issued the SAML token.

Assuming: that each of such functionalities is provided as a service. A data stream architect can study the data stream in order to make a stability prediction of the edited service. Further the designer can study the data stream even to modify the architecture of the service itself; if, for example, the four federated entities are distributed in Europe but two of them are both in Italy, the data stream architect could compare the stability of an architecture where the four entities are seen as four branches of the company, with an architecture where the two entities in Italy are connected through a virtual dedicated LAN resulting in a company with only three branches for a stability point of view. The data stream architect can take the best decision based on a trade-off analysis in terms of total stability of the service versus implementation cost for the chosen solution. While, to evaluate the cost of an architectural solution could be a simple task, to compute the total stability function of a complex data stream is not straightforward. In the next section we present a useful tool that can help the data stream architect performing a stability prediction analysis.

7 Implementation

In this section we first show the main capability of the proposed stability prediction tool and then we have demonstrated the usage of the tool with respect to the case study presented in the previous section.

7.1 The Component

The proposed algorithm was developed as a component for Active BPEL Designer in which is a widely used data stream designing tool. Once installed the plug in allows the data stream designer to perform a stability analysis for a BPEL Wireless Sensor Network Service edit. More precisely it allows to:

- Retrieve the stability function for the data stream under design;
- Evaluate the stability of the data stream at a specific point in time, that is the probability that the data stream will not fail until the specified time;
- Plot the resulting stability function with respect to the time;
- Obtain usual stability metrics, such as the MTTF (Mean Time To Failure), for the analyzed data stream.

In order to perform the above described analysis the designer has to provide the data stream with the information required by the model (such as the symbolic expression of the stability function of each activity, and the transition probabilities). Such information is directly embedded in the BPEL description of the data stream by exploiting the standard WS-BPEL extensibility. This allows the tool to remain compliant with any WS-BPEL edit despite the specific editor adopted for its design. Moreover the component extends the Active BPEL Designer interface to simplify the provisioning of stability related data. Then, in order to compute the total data stream stability function, the component uses an XSLT style-sheet to convert the data stream BPEL/XML based representation to an internal representation that only includes the data stream dependability attributes and the recognized stability methods. After this transformation has been done, the component calls a class that calculates the global stability function as described in the reduction algorithm. The desired analysis estimates are then obtained by evaluating the retrieved stability function. Symbolic operations are made possible by the adoption of the Math Eclipse component.

7.2 Experiment

With respect to the case study data stream depicted in the previous section, the step made by the reduction algorithm is to obtain the data stream stability function. Each step is represented by a numbered box in which the methods recognized by the algorithm are highlighted. In the first and third steps sequence methods are recognized and reduced (dotted boxes). In the second and fifth steps XOR parallel methods are matched and reduced (dashed boxes). In the fourth step an AND-SPLIT configuration is found (dot dashed boxes). Even though pencil and paper calculation is possible, this is for sure an error prone process, as well as a time consuming one, since the most complex is the data stream graph the toughest is to evaluate the stability function by hand. As an alternative the stability function for the data stream can be automatically evaluated by using the proposed component, the data stream as it could be implemented with the Active BPEL Designer tool. To make possible the evaluation of the stability estimates it was required specifying the stability function of each activity as well as the transition probabilities. In our case study, for the sake of simplicity, we assumed all the non-empty activities to have stability functions exponentially distributed with the same failure rate value. We explicitly note that evaluating the failure rate of Wireless Sensor Network Service is beyond the scope of this paper, we instead use a realistic value of 0.048 failures /second as resulting from the experiments. The transition probabilities were assumed to be p= 52% and q = 0.39%, and the value of k for the Synchronizing Parallel method is always equals to the number of branches of the method. Once fixed those values the tool can infer the desired stability evaluations. For example using the component we can easily compare the stability of the two architectural solutions presented in the previous section.

8 Conclusions

In this paper we have proposed a formal approach, this approach allow a data stream architect to perform stability analysis of a Wireless Sensor Network Service-based service. The approach exploits the concept of stability methods to evaluate the stability function of a wide class of data streams. We have integrated our reduction process into an edit engine so to provide a useful component, which can be used to perform a stability analysis for a planned data stream, as well as to compare the stability of alternative solutions in a hypothesis analysis.

9 Acknowledgements

This work has been supported by the National Natural Science Foundation of China (NNSF of China) (10771039) and supported by the Ministry of Science and Technology Innovation Fund Projects (07C26216402072).

References

[1] L. Cherbakov, G. Galambos, R. Harishankar, S. Kalyana, and G. Rackham, *Impact of Service Orientation at the Business Level*, IBM Systems Journal, Vol. 44, No. 4, 2005, pp. 653-668

[2] L. Zeng, B. Benatallah, A. Ngu, M. Dumas, J. Kalagnanam and H. Chang, *QoS-Aware Middleware for Cloud computing Composition*, IEEE Transactions on Software Analysis, Vol. 30, No. 5, May 2004, pp. 311-327.

[3] BIGSF, *Government Web Application Integrity*, The Business Internet Group, San Francisco, 2003.

[4] S. Bhiri, O. Perrin and C. Godart, *Ensuring Required Failure Atomicity of Composite Cloud computing*, Proceedings of the International World Wide Web Conference (WWW'05), Chiba, 2005, pp. 138-147.

[5] *Will Stability Kill the Web Service Composition*? 2010. http://www.cs.rutgers.edu/ rmartin teaching/spring 04/ cs553/papers/009.pdf

---

Hu Ruo , Man, 1968-11, Doctor, Associate professor.E-mail: hu68@163.com, Major Research Areas: Information System Security, Concept Network. The main journal published: 《Computer Science》（Two Articles）,《Computer Application Research》,《Computer Engineering》,《Journal of Tsinghua University》,《Computer Engineering and Applications》.《Special Issue of Sensor Letters》（Two Articles） (SCIE),《Applied Mechanics and Materials》(EI)