（ 続紙 1 ）

京都大学	博士（社会情報学）	氏名	Kemas Muslim Lhaksmana (けます むすりむ らくすまな)
論文題目	Multiagent Systems for Robust IoT Services （頑健な IoT サービスのためのマルチエージェントシステム）		

（論文内容の要旨）

The goal of this research is to provide multiagent systems (MAS) design and implementation methods for developing IoT services that are robust against changes and failures. The main contributions of this research are: (i) proposing a robust topology for the network of IoT services; (ii) introducing a self-organizing MAS modeling method based on role notion for designing flexible IoT services; and (iii) proposing a programming language for implementing flexible IoT services that supports role model implementation.

Chapter 1 provides the overview of this thesis by introducing research issues and objectives, summarizing the proposed solutions, and addressing the thesis outline.

Chapter 2 addresses the background of this thesis by reviewing previous research in IoT, especially the effort to adopt service-oriented computing (SOC) principles and technologies into IoT as Internet of Services (IoS). SOC principles and technologies have been proposed to provide solutions for composing heterogeneous smart objects in IoT services. In addition, this chapter also examines related work in MAS, including self-organization mechanisms in MAS and some existing engineering methodology to support self-organizing MAS development. These are addressed to explore state-of-the-art solutions that would be useful for designing flexibility in robust IoT services.

Chapter 3 proposes an IoT services topology that is robust against failure. In an IoT application, the dynamic nature of IoT services and the dependency between them are the factors that potentially cause cascading failure. This type of failure occurs when failures on one or more IoT services cause failures on the other dependent IoT services. To propose IoT services topology that is robust against cascading failures, this chapter firstly analyzes how such failures propagate in IoT services. To this end, the interdependency among IoT services is modeled as a service network, i.e. an AND/OR graph model of IoT services whose nodes are services and links are dependencies between services. The simulation result shows significant tolerance improvement can be achieved by increasing the number of substitutable services when the number is currently low, e.g. at most 3 substitutable services on average. The findings in this chapter suggest that robust IoT services should be achieved by (i) providing sufficient number of substitutable services, and (ii) embedding *failover flexibility* on composite services, i.e. the capability to perform failover actions by switching among

substitutable services when one or more component services fail. In addition, the simulation result shows the service networks having scale-free topology has better tolerance than that of exponential and random topology. This is contrast to power grids where random topology is better than scale-free.

Chapter 4 proposes a role-modeling method to design *functional flexibility* in IoT services as self-organizing MAS. In contrast with failover flexibility that requires substitutable services, functional flexibility focuses on the capability of an IoT service to change its functionality at runtime to respond against changes or failures. Self-organizing MAS is suitable to represent the collection of flexible IoT services as agents since they share common characteristics, such as autonomous, adaptive, decentralized, etc. However, designing flexible IoT services as self-organizing MAS requires a modeling method that separates between designing agent behavior and behavior adaptation. The separation improves maintainability and provides designers a convenience way to design agent behavior adaptation. To this end, a role-modeling method is proposed, which is a method where each role represents a collection of agent behaviors so that agents can change their behaviors by changing roles. Such feature has not been supported by existing MAS engineering methodologies since they tend to mix agent behaviors and the way agent responding to changes and failures. An IoT-based traffic intersection system is used as a case study that demonstrates the design of some flexible IoT services using the proposed role-modeling method.

Chapter 5 proposes a programming language to implement failover and functional flexibility of IoT services as adaptive agents in MAS. The use of agent-oriented language would be convenience to implement flexible IoT services as agents since agent-oriented properties, such as agent classes, goals, and actions, can be mapped and implemented directly in the program. However, most of existing agent-oriented languages do not provide high-level notion to implement agent's roles. Therefore, a language is proposed to allow IoT services flexible capability to be implemented as high-level agent behavior using role notion. The language is defined by extending AgentSpeak, which originally follows BDI architecture in that agent mental states are defined in terms of goals, beliefs, and plans. The extension parts are converted to AgentSpeak so that Jason interpreter can interpret them. The proposed language allows these mental states to be defined in each role such that agents can adopt and release roles at runtime, according to role transition models. The language also has better support on modularity since it is capable of handling plans having the same triggering events that are defined either in the same or in different files. The current Jason does not handle plans with the same triggering event; only one plan that can be executed, whereas the other plans become useless. An implementation of a case study demonstrates that the modular

implementation of roles does not significantly increase the number of non-commented lines of code. This shows that the modularity can be increased while maintaining readability.

Chapter 6 concludes this thesis by summarizing the original contributions of the proposed solutions for supporting the development of robust IoT services. Possible future directions that deal with reliability issue in IoT systems are also provided.

注)論文内容の要旨と論文審査の結果の要旨は１頁を３８字×３６行で作成し、合わせて、３,０００字を標準とすること。

論文内容の要旨を英語で記入する場合は、４００～１,１００words で作成し

審査結果の要旨は日本語５００～２,０００字程度で作成すること。

（続紙　２　）

（論文審査の結果の要旨）

本論文は、システムの変更や障害に頑健な IoT サービスの設計および実装手法の確立を目的とするものである。IoT サービスの依存関係を示すネットワークを分析し、環境の変化に適応する IoT サービスのモデリング手法とそのモデルを実装するためのプログラミング言語を提案している。

1. IoT サービスのネットワーク分析

IoT サービスは、センサーなどのデバイスをサービス化し、それらのサービスの連携によって新たなサービスを構築するものである。したがって、IoT サービスの障害は、そのサービスを利用する IoT サービスの障害を連鎖的に引き起こす。そこで本研究では、まず、IoT サービス間の依存関係を表すネットワーク上で障害の伝播を分析している。具体的には、スケールフリーネットワーク、指数関数的ネットワーク、ランダムネットワークという３種の異なるネットワーク構造に対して、連鎖的な障害のシミュレーションを行った。その結果、いずれのネットワーク構造であっても、代替可能なサービス数が増加するにつれて、連鎖的な障害の影響を受けるサービスが指数関数的に減少することを示している。

2. IoT サービスのためのロールモデリング

代替可能な IoT サービスが確保できない場合に連鎖的な障害を防ぐには、IoT デバイスの機能や役割を動的に変更して代替サービスとする必要がある。そこで、自律性と適応性に優れたマルチエージェントシステムの手法を導入する。具体的には、エージェントの機能と役割の記述を分離するロールモデリングを導入する。この分離は、環境に適応して機能を変える IoT サービスの設計を容易にする。ロールモデリングでは IoT サービスの機能の集合をロールとして定義し、環境に合わせたロールの変更を記述できる。本手法を用いて交通制御システムの設計を行い、車感センサーの障害時に、交通量流量を計測するカメラセンサーで代替することで、提案したモデリングの有効性を示している。

3. IoT サービスのためのエージェント指向プログラミング

従来のプログラミング言語では、エージェントのロールを記述する手段を提供していない。そこで、IoT サービスの機能と役割を分離して実装する言語を提案している。具体的には、既存のエージェント指向プログラミング言語を拡張し、ロールごとにエージェントの機能を定義し、エージェントには環境に応じたロールの選択および解除を記述する。また、このような拡張記述を既存のエージェント指向プログラミング言語に変換するトランスレータを開発している。実際にロールモデリングを用いて設計した交通制御システムを実装することで、提案したプログラミング言語の記述性を検証している。

以上、本論文は、IoT サービスにおける連鎖的な障害を分析し、耐障害性に優れた IoT サービスのモデリング手法と、それを実装するプログラミング言語を提案し、頑健な IoT サービスの構築に寄与している。よって、本論文

は博士（情報学）の学位論文として価値あるものと認める。また、平成 28 年 7 月 28 日に実施した論文内容とそれに関連した試問の結果、合格と認めた。

注)論文審査の結果の要旨の結句には、学位論文の審査についての認定を明記すること。

更に、試問の結果の要旨（例えば「平成　年　月　日論文内容とそれに関連した

口頭試問を行った結果合格と認めた。」）を付け加えること。

Web での即日公開を希望しない場合は、以下に公開可能とする日付を記入すること。

要旨公開可能日：　　年　　月　　日以降