# A Semantic-Based Malware Detection System Design Based on Channels

Peige Ren, Xiaofeng Wang, Chunqing Wu, Baokang Zhao, Hao Sun



# A Semantic-based Malware Detection System Design Based on Channels

Peige Ren, Xiaofeng Wang, Chunqing Wu, Baokang Zhao, Hao Sun

School of Computer Science
National University of Defense Technology
Changsha, Hunan, CHINA

renpeige@163.com, {xf_wang,chunqingwu,bkzhao}@nudt.edu.cn
sunhao4257@gmail.com

**Abstract.** With the development of information technology, there are massive and heterogeneous data resources in the internet, as well as the malwares are appearing in different forms, traditional text-based malware detection cannot efficiently detect the various malwares. So it is becoming a great challenge about how to realize semantic-based malware detection. This paper proposes an intelligent and active data interactive coordination model based on channels. The coordination channels are the basic construction unit of this model, which can realize various data transmissions. By defining the coordination channels, the coordination atoms and the coordination units, the model can support diverse data interactions and can understand the semantic of different data resources. Moreover, the model supports graphical representation of data interaction, so we can design complex data interaction system in the forms of flow graph. Finally, we design a semantic-based malware detection system using our model; the system can understand the behavior semantics of different malwares, realizing the intelligent and active malware detection.

**Keywords.** Malware detection; interactive coordination model; semantic

## 1 Introduction

With the rapid development of Internet technology, the data types and data amount in the internet is growing in amazing speed, and more and more malwares with different types are appearing, such as virus, worm and Trojan horse. So it is becoming a great challenge to accurately detect the various malwares from massive and heterogeneous internet data resources.

Traditional malware detection methods are based on text-based feature codes matching [1, 2], they cannot realize semantic-based similarity matching. And the malwares are appearing in different forms, it is difficult to accurately detect the various malwares from massive and heterogeneous internet data resources. This paper presents an intelligent and active data interactive coordination model based on channels, and we designed a semantic-based malware detection system using our model.

Specifically, the contributions of this paper are fourfold: (1) Abstract the behaviors of data transmission, organization and processing as coordination channel, coordination atom and coordination unit respectively, which can support the understanding of semantic and data initiatively push, realize diverse data interactions; meanwhile define complex control functions during the data interaction, supporting the modeling of intelligent, initiative and flexible data interaction systems. (2) Support graphical representation of data interactions, which can be used to explicitly and visually design complex data interaction systems in the form of flow graph. (3) Accurately define the behavioral semantics of coordination channels and coordination atoms with logical mathematical formulas, which can be used to strictly verify the consistency between the system model design and the system realization. (4) Design a semantic-based malware detection system using the proposed model, which can realize semantic-based intelligent malware detection.

The remainder of the paper is organized as follows. Section 2 introduces the related work. Section 3 presents the intelligent and active data interactive coordination model based on channels. Then we present the coordination channels and coordination atoms, introducing their classification, operations and behavior semantics in section 4 and section 5. Followed the coordination unit are presented, and we design several special coordination units in section 6. In section 7, we design a semantic-based malware detection system using our model and conclusion can be found in Section 8.

## 2 Related Works

Future data interaction system should have the characteristics of intelligence, initiative and flexibility. However, for the theoretical modeling, we have not found a special data interactive coordination model to design data interaction system effectively. The related research works about interactive coordination mainly pay attention to the fields of multi-Agent and software coordination.

Multi-Agent interaction is the kernel of the research of distributed artificial intelligence and Multi-Agent Systems (MAS) [3], it realized the capability of autonomic group interaction between multiple Agent, can solve a complicated problem coordinately in a distributed environment. Early researches about Multi-Agent interaction include typical distributed problem solving system such as Actor [4], DVM [5], and MACE [6], these works emphasize the compact group cooperation between the Agent, and data interaction is in the form of tight coupling between entities. Researchers later recognized that the tight coupling interaction collaborative cannot meet the demand of increasingly complex network environment and the reality needs, which prompting a variety mode of interaction between agents. A BDI language called MAL is presented in [7], it overcomes the misunderstanding of the concepts of belief, desire and intention, supporting multi-agent interaction more effectively.

On the other hand, the coordination between software entities has been a hotspot problem. Software coordination [8] means that establishing connections between software entities, constraining the interaction behaviors between them to make them work together in harmony in an open environment. Farhad Arbab divided the software

coordination models into data driven coordination model [10] and control driven coordination model [9]. The data driven coordination model mainly focused on the data exchange between coordination entities, realizing the shared space-based anonymous communication between them, the coordination entities need to call coordination primitives to exchange information with outside. The control driven coordination model, such as Manifold [11], Darwin [12], IWIM [13] and Reo [14], mainly focused on the status change of coordination entities and the control flow between them. The coordination entities were seen as black boxes with well-defined interfaces. They can perceive the change of external environment by accepting the messages through interfaces, which then caused their status change; meanwhile send messages to external environment to change surrounding environment. The control driven coordination model realized the separation of computation and coordination, helping to realize the maintenance and reuse of computing and coordination module.

## 3 The Overview of the Model

For realizing the intelligent, active and flexible data interaction in the internet, we abstracted the functions of data interaction, proposed a kind of intelligent and active data interactive coordination model.

In the model, the functional modules of data transmission, organization and control are defined abstractly as the coordination channel, coordination atom and coordination unit respectively. The coordination channels are the basic construction unit of this model, which can realize various data transmissions. The coordination atoms are the management units of the coordination channels as well as the data organization units in the network, which can be divided into syntax and semantic coordination atoms. The coordination atoms can find the useful data resources intelligently, and connect the corresponding channels together to form a data channel, realizing intelligent data aggregation, organization and distribution. The coordination units are formed by some coordination atoms and coordination channels connected in a certain topological structure, they can realize some specific data control function during data interaction.

## 4 Coordination Channels

A coordination channel can be seen as a point-to-point medium of communication between two interactive interfaces, it can transmit data resources. Every channel has two channel ends, which can be divided as three types: send ends, receive ends and bidirection ends. A send end accepts data into channel and sends data to the receive end along the channel; a receive end accepts data from channel and send data outside; a bidirection end can realize the functions of both send channel and receive channel.

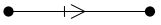
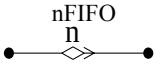
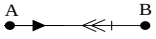
The coordination channels are the basic functional unit in the model. They can be assigned by different functions according to the actual requirements. The synchronous channel supports the synchronous data transmission between its ends, can realize the real-time synchronous data interaction between users; while the asynchronous channel

can cache the data inside the channel, can realize the loosely coupled asynchronous interaction between users, avoiding the interdependence between the users.

### 4.1 Coordination Channel Types

The coordination channels can be divided into data flow channels and control channels. The data flow channels include: Sync channel, SyncWrite channel, AsyncWrite channel, FIFO channel, RAW channel, etc. The control channels transmit only the control messages, mainly used to realize the remote procedure call between the entities connected the channel. Every channel has two ends of the same type or not. The behavior of a channel depends on its synchronizing properties, the types and numbers of its ends, the size of buffer inside the channel, and the loss policy, etc. Table 1 shows the types and the behavior description of channels. Certainly, we can present more new channels according to our requirement.

**Table 1. Coordination channel types**

| Channel Type | | Channel Behavior Description |
|---|---|---|
| Data Flow Channel | Sync  | The channel has a send and a receive end, the I/O operations on the ends must succeed at the same time. |
| | SyncLossy  | The channel has a send and a receive end, the send end can accepts data resources from outside at any time, if the operations on the receive end cannot take the data simultaneously, the data is lost. |
| | SyncWrite  | The channel has two send ends, the write operations on its two ends must succeed simultaneously, and the accepted data objects are lost. |
| | AsyncWrite  | The channel has two send ends, the write operations on its two ends must succeed asynchronously, and the accepted data objects are lost. |
| | Filter(pat)  | The channel has a send and a receive end, when the data written to the send end does not match with the pattern pat, the data is lost; or else the channel behaves the same way as a Sync channel. |
| | FIFO  | The channel has a send, a receive end and an unbounded buffer, the receive end can accept data at any time and the data are stored in the buffer, the operations on the receive end can obtain the data in the FIFO order. |
| | nFIFO n  | The channel has a send, a receive end and a bounded buffer with capacity n, it operate in the same way of the FIFO channel until the buffer is full. |
| | RAW   | The channel has two bidirection ends, the operation on end A can only obtain data from B when the data written to A is obtain by the operation on B simultaneously. |
| Control  | | The channel has a send and a receive end, the user connected to send end can send control command to realize the remote procedure call between the users. |

### 4.2 Behavior Semantics of Coordination Channels

This section tries to describe the behavior semantics of coordination channels in formula. For a channel c, whose send and receive end are $c_i$ and $c_o$, we have:

**recv($c_i$, d)** denotes that the data object d is successfully written to the channel end $c_i$. Particularly, **syn_recv($c_i$, d)** means that the data d is successfully written to the sync coordination channel end $c_i$. While **offer($c_o$, p)** denotes the multi-set of pairs ($c_o$, d), d is a data object that taken from the channel end $c_o$ and match with the pattern p.

We use * denotes a channel end of any other channel, $\hat{e}$ denote the unique coordination atom on which the channel end e coincides, $d \ni p$ means data d matches with the pattern p, recv(ê,d) and offer(ê,p) express the data operating on coordination atom $\hat{e}$ (see section 5.3). So we can define the behavior semantics of coordination channels with logical mathematical formulas, the following are some examples:

**Sync channel behavior semantic:** For a Sync channel c, $c_i$ and $c_o$ are its send and receive end, the behavior semantic of Sync channel can be defined by (1) and (2).

$$\text{syn\_recv}(c_i, d) = \text{syn\_recv}(\widehat{c_o}, d) \wedge (d \ni p) \tag{1}$$

$$\text{offer}(c_o, p) = \{(c_o, d) \mid (*, d) \in \text{offer}(\widehat{c_i}, p)\} \tag{2}$$

**AsyncWrite channel behavior semantic:** The behavior semantic of AsyncWrite channel c whose send ends are $c_{i1}$ and $c_{i2}$ can be defined by equations (3) and (4).

$$\text{recv}(c_{i1}, d_1) = \text{recv}(c_{i2}, d_2) = \text{true} \tag{3}$$

$$|\text{offer}(\widehat{c_{i1}}, p_1)| \neq |\text{offer}(\widehat{c_{i2}}, p_2)| \tag{4}$$

**Filter(pat) channel behavior semantic:** The behavior semantic of a Filter(pat) channel c whose send and receive end are $c_i$ and $c_o$ can be defined by (5) and (6).

$$\text{recv}(c_i, d) = d \not\ni pat \vee \text{recv}(\widehat{c_o}, d) \tag{5}$$

$$\text{offers}(c_o, p) = \{(c_o, d) \mid (*, d) \in \text{offers}(\widehat{c_i}, p) \wedge d \ni pat\} \tag{6}$$

## 5 Coordination Atoms

Coordination atoms are the organization function module of channels, as well as the data management module in the network. The coordination atoms can accurately find the right data resources and actively connect the correlative channels together to form data transmission path, without the requirements of the address of interaction parties, realizing the space decoupling between the interaction parties.

### 5.1 Coordination Atom Types

The coordination atoms can be divided into syntactic atoms and semantic atoms.

The syntactic atoms organize the channels according to the data description forms, realizing the aggregation, organization and forwarding of data resources in the same

representation form. We denote a syntactic atom by the symbol ○, as shown in Figure 1. The syntactic atoms are divided into several types. Here, we define the set of all channel ends coincident on an atom N as [N]={x | x→N}, and divide the [N] into the sets Sed(N) , Rev(N) and Bir(N), respectively denoting the sets of send, receive and bidirection ends that coincide on N. If Sed(N)≠∅∧Rev(N)=∅, N is called a send coordination atom, such as A1 and A2 in Figure 1; if Sed(N)=∅∧Rev(N)≠∅, N is called a receive coordination atom, such as A5 and A6; if Bir(N) ≠∅, N is a called bidirection coordination atom; and if Sed(N)≠∅∧Rev(N)≠∅, N is called mixed coordination atom, such as A3 and A4.

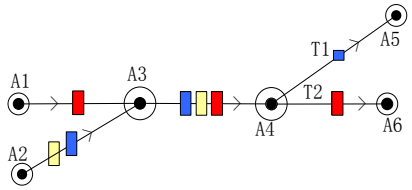



**Fig.1.** Syntactic coordination atom

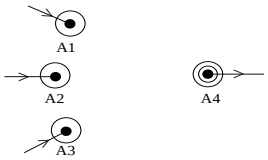



**Fig.2.** Semantic coordination atom

The semantic coordination atom, denoted by the symbol ◎ (as shown in Figure 2), can extract the semantic information from various data resources. The semantic coordination atom first abstract the semantic features of data resources to construct a high-dimensional feature space, and the data resources and user request are expressed as high-dimensional points in the feature space, then use similarity search methods to find the data resources that have similar semantic to the user request, and forward them to the users. In Figure 2, the semantic coordination atom A4 can distinguish the semantic of the data resources from A1, A2 and A3, select the data resources with similar semantic to the user request and forward them.

### 5.2 Coordination Atom Operations

The main coordination atom operations are shown in Table 2. The parameter t indicates a time-out value, the operations fail if it does not succeed within the time t.

**Table 2. Coordination atom operations**

| Operations | Behavior Description |
|---|---|
| a_create(A) | Create a coordination atom A for the connected channels |
| a_merge(A1, A2, A) | Merge the coordination atoms A1 and A2 to form a new atom A |
| a_split(A, A1(pat)) | Produce a new atom A1 according to the data pattern pat, the channel ends matched with pat are split from A and connected to A1. |
| a_syn_write([t], A, d) | The operation succeeds as soon as having written the data object d to every channel end x∈[A] simultaneously |
| a_write([t], A, d) | The operation succeeds when the data d is written to every channel end x∈[A] |
| a_read([t], A, v, p) | Read a data compatible with the pattern p from any one channel ends x∈[A] into the variable v. |

| | |
|---|---|
| a_take([t], A, v, p) | Take a data compatible with the pattern p from any one channel ends x ∈[A] and read it into the variable v. |
| a_alert(A, p, f) | Register the function f as the callback function of the data compatible with pattern p in coordination atom A |

### 5.3 Behavior Semantics of Coordination Atoms

The coordination atom manages the channel ends coinciding on it, and the behavior semantic of a coordination atom is the integration of the behavior semantics of all the channel ends on it, describing the data distribution on it.

For coordination atom A and a data pattern p, and the predicate $\int$ (O) designates an operation O is pending on its respective coordination atom if it is true, we define:

$$\mathrm{offer}(\mathrm{A},\mathrm{p}) = \begin{cases} \bigcup_{\int(\mathrm{a\_syn\_write}(\mathrm{A,d}))\vee(\mathrm{a\_write}(\mathrm{A,d}))} \{\langle \varepsilon, d\rangle \mid d \ni p\} & \text{A is send atom} \\ \bigcup_{c_o \in Rev(A)} \mathrm{offer}(\mathrm{c_o},\mathrm{d}) & \text{otherwise} \end{cases} \quad (7)$$

If offer (A, p) is empty, we cannot obtain data from A; if offer (A, p) is not empty, that is, $\exists\langle \mathrm{x},\mathrm{d}\rangle \in \mathrm{offer}(\mathrm{A},\mathrm{p})$, then we can obtain data from A. The symbol ε represents "no channel end", means that when A is a send coordination atom, the data can only be obtained from the write operations pending on A.

For a coordination atom A and a data d, we define:

$$\mathrm{recv}(\mathrm{A},\mathrm{d}) = \begin{cases} \bigcap_{\int(\mathrm{a\_take(A,v,p)})\vee(\mathrm{a\_alert(A,p,f)})} \mathrm{d} \ni \mathrm{p} & \text{A is receive atom} \\ \bigcap_{c_i \in \mathrm{Sed(A)} \wedge \mathrm{d} \ni \mathrm{p}} \mathrm{recv}(c_i,\mathrm{d}) & \text{otherwise} \end{cases} \quad (8)$$

From the equation (8) , we can find that when A is a receive coordination atom, it accepts the data d only if d matches with the pattern p of all a_take and a_alert operations pending on A; otherwise, A accepts d only when all send ends in [N] accept d.

For a coordination atom A that connected channels are all Sync channels, we have:

$$\mathrm{syn\_recv}(\mathrm{A},\mathrm{d}) = \begin{cases} \bigcap_{\int(\mathrm{a\_take(A,v,p)})\vee(\mathrm{a\_alert(A,p,f)})} \mathrm{d} \ni \mathrm{p} & \text{A is receive atom} \\ \bigcap_{c_i \in \mathrm{Sed(A)} \wedge \mathrm{d} \ni \mathrm{p} \wedge \mathrm{c} \ni \mathrm{Sync}} \mathrm{syn\_recv}(c_i,\mathrm{d}) & \text{otherwise} \end{cases} \quad (9)$$

The equation (9) has the similar behavior semantic as equation (8), except that the operations on A must be done simultaneously.

For a mixed atom A, we have:

$$\tau(\mathrm{A}) = \{<\mathrm{c_x},\mathrm{d}> \mid <\mathrm{c_x},\mathrm{d}> \in \mathrm{offer}(\mathrm{A},\mathrm{p}) \wedge \mathrm{recv}(\mathrm{A},\mathrm{d}) \wedge (\mathrm{d} \ni \mathrm{p})\} \quad (10)$$

In the equation (10), τ (A) means the data objects that are eligible for transfer at the mixed coordination atom A.

## 6 Coordination Units

A coordination unit is formed by a set of coordination channels organized in special topology to realize specific control function during the data interaction in the network.

With coordination units, the actors of data interaction do not need to think about how to control the data flow reasonably, realizing intelligent and flexible data interaction. Here we list several coordination units with special functions as follows. Besides, we can design more various flexible coordination units according the system requirement.

**Data flow controller:** This kind of coordination units can monitor and control the data flow in the network. As shown in the left of Figure 3, the unit is formed by the FIFO channels T1, T2, T3 and a synchronous channel T4. The data flow from channel ends a and b to end c is controlled by the channel T4, only when a data item is taken from end d synchronously, a data item can flow into channel T3 from A. The taking operation on end d can monitor and control the data flow from A to end c. While in the right of Figure 3, T4 is an nFIFO channel with a buffer of size n. Operations on the channel end d can monitor, back up the data flow to the end c, and the channel T4 can be seen as a leaky bucket policer to adjust the transmission rate of the data flow.

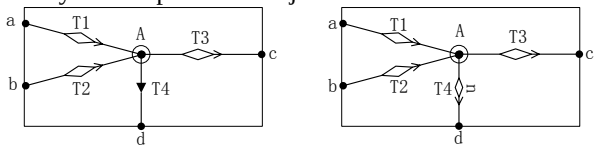



**Fig. 3.** Data Flow Controller

**Semantic Aggregator**: There have been many different data resources with similar semantic but of different representations, traditional syntactic-based matching methods cannot discover the data resources of user requirement efficiently. For realizing the accurately and efficiently discovery of data resources, we proposed the semantic aggregator. As shown in Figure 4, the semantic aggregator are formed by three Sync channels, one RAW channel, three syntactic coordination atoms and one semantic coordination atom. The data resources input from the channel ends a, b and c may have different types, to realize the semantic-based data retrieval, the semantic coordination atom D abstracts the semantic features of data resources and the user request from channel end d to map the data resources and user request to a high-dimensional feature space, then use similarity search methods to find the data resources that have similar semantic to the user request, and answer the user with the data resources through channel end d.

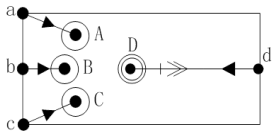



**Fig. 4.** Semantic Aggregator

## 7 Semantic-based Malware Detection System

With the development of information technology, there are various malware in the internet, such as worm, Trojan horse and zombie. They are in different types, but all

are harmful to the internet environment. Traditional text-based feature codes matching cannot detect all the malwares in different types. To realize semantic-based malware detection, we can use the semantic aggregator to distinguish the malware from normal data resources according to their semantics.

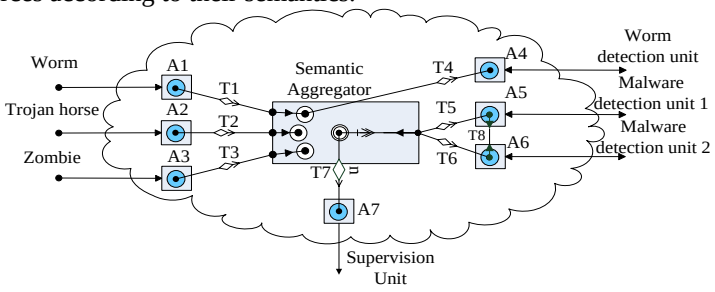



**Fig. 5.** The Semantic-based Malware Detection System

As shown in Figure 5, we design a semantic-based malware detection system. The worm, Trojan horse and zombie are different in data type, but all belong to the malware. The system built the data path between worm and worm detection unit according to syntactic-based accurate matching; the data resources flowed in the path are of the same type. The system built the data path connected to malware detection unit 1 and 2 using the semantic aggregator, the semantic aggregator can distinguish malware in this system, and the worm, Trojan horse and zombie can flow to malware detection unit 1 and 2 automatically. The coordination channel between coordination atom A5 and A6 is a SyncWrite channel, it provides that the malware detection unit 1 and 2 must obtain the malware synchronously. Besides, the coordination channel T7 is an nFIFO channel connected to the supervision unit, realizing the adjustment of the data transmission rate, and the supervision unit can monitor and back up the data resources flowed to the malware detection unit 1 and malware detection unit 2.

## 8 Conclusion

In order to realize semantic-based malware detection and data interaction efficiently, we propose an intelligent, active and flexible interactive coordination model based on channels. We present the coordination channels, coordination atoms, and coordination units in the model. The model can describe the process of the data organization, transmission and processing in network clearly, and support the graphical expression of data interaction. And we accurately define the behavioral semantics of coordination channels and coordination atoms, which can strictly verify the consistency between the system model design and the system realization. Finally, a semantic-based malware detection system instant is designed using the model. The model can efficiently organize, transmit and process the data resources in open, dynamic and heterogeneous network environment, which can promote the development of the advanced data interaction mechanisms.

## Acknowledgment

The work described in this paper is partially supported by the grants of the National Basic Research Program of China (973 project) under Grant No.2009CB320503, 2012CB315906; the project of National Science Foundation of China under grant No.61070199, 61103189, 61103194, 61103182, 61202488, 61272482; the National High Technology Research and Development Program of China (863 Program) No.2011AA01A103, 2012AA01A506, 2013AA013505, the Research Fund for the Doctoral Program of Higher Education of China under Grant No.20114307110006, 20124307120032, the program for Changjiang Scholars and Innovative Research Team in University (No.IRT1012), Science and Technology Innovative Research Team in Higher Educational Institutions of Hunan Province ("network technology"); and Hunan Province Natural Science Foundation of China (11JJ7003).

## References

1. Filiol E, Helenius M, Zanero S. Open problems in computer virology. J. Comput. Virol. 1, 2006, 55-66.
2. Szor P. The art of computer virus research and defense. Addison-Wesley Professional, 2005.
3. Genesereth MR., Ketchpe. S. Software Agents. Communications of the ACM, 1994, 37(7): 48-96.
4. Hewitt C. Viewing Control Structures as Patterns of Passing Messages. Artificial Intelligence, 1977, 8(3): 264-323.
5. Lesser V, Corkill. D. Functionally accurate: Cooperative distributed systems. IEEE Transactions on Systems, Man, and Cybernetics part C, 1981, 11(1): 81-96.
6. Durfee EH, Montogomery TA. MICE: A flexible testbed for intelligent coordination experiments. Proceedings of Distributed artificial intelligence workshop, 1988.
7. KANG XQ, SHI CY. MULTI-AGENT INTERACTION BASED ON BDI. CHINESE JOURNAL OF COMPUTERS, 1999, 22(11): 1166-1171(in Chinese).
8. LV J, MA XX, TAO XP, XU F, HU H. The research and development of the Internetware. Science in China (Series E: Information Sciences), 2006, 36(10):1037~1080(in Chinese).
9. Papadopoulos G A, Arbab F. Coordination models and languages. Software Engineering (SEN) SEN-R9834, Centrum voor Wiskunde en Informatica (CWI), Amsterdam, Netherlands, Dec. 1998.
10. Kielmann T. Objective Linda: A Coordination Model for Object-Oriented Parallel Programming [Ph.D. Thesis]. Germany: University of Siegen, 1997.
11. Arbab F, Herman I, Spilling P. An overview of manifold and its implementation. Concurrency and Computation: Practice and Experience-CONCURRENCY, 1993, 5(1):23-70.
12. Magee J, Dulay N, Eisenbach S, Kramer J. Specifying distributed software architectures. European Software Engineering Conference, 1995. 137-153.
13. Arbab F. IWIM: A communication model for cooperative systems. Proceedings of the 2nd International Conference on the Design of Cooperative Systems, 1996.
14. Arbab F. Reo: a channel-based coordination model for component composition. Mathematical Structures in Computer Science - MSCS, 2004, 14(3):329-366.