7
Referências Bibliográficas

[1] MADDEN, S. R. et al. Tinydb: an acquisitional query processing system for sensor networks. **ACM Trans. Database Syst.**, ACM, New York, NY, USA, v. 30, n. 1, p. 122–173, 2005. ISSN 0362-5915. 1.3, 3.1.2

[2] NEWTON, R.; MORRISETT, G.; WELSH, M. The regiment macroprogramming system. In: **IPSN '07: Proceedings of the 6th international conference on Information processing in sensor networks**. New York, NY, USA: ACM, 2007. p. 489–498. ISBN 978-1-59593-638-X. 1.3, 3.1.1, 3.1.2

[3] AWAN, A.; JAGANNATHAN, S.; GRAMA, A. Macroprogramming heterogeneous sensor networks using cosmos. **EuroSys '07: Proceedings of the 2nd ACM SIGOPS/EuroSys European Conference on Computer Systems 2007**, ACM, New York, NY, USA, p. 159–172, 2007. 1.3, 3.1.2

[4] LEVIS, P.; CULLER, D. Maté: a tiny virtual machine for sensor networks. In: **ASPLOS-X: Proceedings of the 10th international conference on Architectural support for programming languages and operating systems**. New York, NY, USA: ACM, 2002. p. 85–95. ISBN 1-58113-574-2. 1.3

[5] HUI, J. W.; CULLER, D. The dynamic behavior of a data dissemination protocol for network programming at scale. In: **Proceedings of the 2nd international conference on Embedded networked sensor systems**. New York, NY, USA: ACM, 2004. (SenSys '04), p. 81–94. ISBN 1-58113-879-2. Disponível em: <http://doi.acm.org/10.1145/1031495.1031506>. 1.3

[6] MUNAWAR, W. et al. Dynamic tinyos: Modular and transparent incremental code-updates for sensor networks. In: **Proceedings of the IEEE International Conference on Communications (ICC)**. Cape Town, South Africa, May 23-27: [s.n.], 2010. 1.3

[7] KASTEN, O.; RöMER, K. Beyond event handlers: programming wireless sensors with attributed state machines. In: **IPSN '05: Proceedings of the**

4th international symposium on Information processing in sensor networks. Piscataway, NJ, USA: IEEE Press, 2005. p. 7. ISBN 0-7803-9202-7. 1.3

[8] BISCHOFF, U.; KORTUEM, G. A state-based programming model and system for wireless sensor networks. In: **PERCOMW '07: Proceedings of the Fifth IEEE International Conference on Pervasive Computing and Communications Workshop**. Washington, DC, USA: IEEE Computer Society, 2007. p. 261 –266. ISBN 0-7695-2788-4. 1.3

[9] HAREL, D. Statecharts: A visual formalism for complex systems. **Sci. Comput. Program.**, Elsevier North-Holland, Inc., Amsterdam, The Netherlands, The Netherlands, v. 8, n. 3, p. 231–274, 1987. ISSN 0167-6423. 1.3

[10] HAREL, D.; NAAMAD, A. The statemate semantics of statecharts. **ACM Trans. Softw. Eng. Methodol.**, ACM, New York, NY, USA, v. 5, n. 4, p. 293–333, 1996. ISSN 1049-331X. 1.3

[11] LEVIS, P. et al. Tinyos: An operating system for sensor networks. In: **in Ambient Intelligence**. [S.l.]: Springer Verlag, 2004. 2, 3.2.1, 4.3

[12] GAY, D. et al. The nesc language: A holistic approach to networked embedded systems. ACM, New York, NY, USA, p. 1–11, 2003. 2, 4.3

[13] WIKIPEDIA. **List of wireless sensor nodes — Wikipedia, The Free Encyclopedia**. February 2011. [Online; accessed 13-March-2011]. Disponível em: <http://en.wikipedia.org/w/index.php?title=List_of_wireless_sensor_nodes&oldid=414463291>. 2.2.1

[14] LEVIS, P. et al. Tossim: accurate and scalable simulation of entire tinyos applications. In: **Proceedings of the 1st international conference on Embedded networked sensor systems**. New York, NY, USA: ACM, 2003. (SenSys '03), p. 126–137. ISBN 1-58113-707-9. Disponível em: <http://doi.acm.org/10.1145/958491.958506>. 2.2.1, 4.1, 4.3, C

[15] BRANCO, A.; RODRIGUEZ, N. Macroprogramação em rede de sensores sem fio. In: **Monografias em Ciência da Computação**. [S.l.]: PUC-Rio, 2010. v. 02/2010. 3.1, 3.1.2

[16] CERVANTES, H.; DONSEZ, D.; TOUSEAU, L. An architecture description language for dynamic sensor-based applications. In: **Consumer Communications and Networking Conference, 2008. CCNC 2008. 5th IEEE**. [S.l.: s.n.], 2008. p. 147–151. ISSN 0197-2618. 3.1.1, 3.1.2

[17] KOTHARI, N. et al. Reliable and efficient programming abstractions for wireless sensor networks. **PLDI '07: Proceedings of the 2007 ACM SIGPLAN conference on Programming language design and implementation**, ACM, New York, NY, USA, p. 200–210, 2007. 3.1.1, 3.1.2

[18] NEWTON, R.; WELSH, M. Region streams: functional macroprogramming for sensor networks. In: **DMSN '04: Proceeedings of the 1st international workshop on Data management for sensor networks**. New York, NY, USA: ACM, 2004. p. 78–87. 3.1.2

[19] BAKSHI, A. et al. The abstract task graph: a methodology for architecture-independent programming of networked sensor systems. **EESR '05: Proceedings of the 2005 workshop on End-to-end, sense-and-respond systems, applications and services**, USENIX Association, Berkeley, CA, USA, p. 19–24, 2005. 3.1.2

[20] LIN, K.; LEVIS, P. Data discovery and dissemination with dip. In: **IPSN '08: Proceedings of the 7th international conference on Information processing in sensor networks**. Washington, DC, USA: IEEE Computer Society, 2008. p. 433–444. ISBN 978-0-7695-3157-1. 4.1.1

[21] FONSECA, R. et al. **The Collection Tree Protocol (CTP) - TEP 123**. http://www.tinyos.net/tinyos-2.1.0/doc/html/tep123.html, 02 2007. Disponível em: <http://www.tinyos.net/tinyos-2.1.0/doc/html/tep123.html>. 4.1.1

[22] CROSSBOW. **MICAz datasheet**. 2004. Product folder. Disponível em: <http://www.xbow.com/asset-tracking/support/knowledge-base.html>. 4.3, 4.5.1

[23] SHNAYDER, V. et al. Simulating the power consumption of large-scale sensor network applications. In: **Proceedings of the 2nd international conference on Embedded networked sensor systems**. New York, NY, USA: ACM, 2004. (SenSys '04), p. 188–200. ISBN 1-58113-879-2. Disponível em: <http://doi.acm.org/10.1145/1031495.1031518>. 5.2

A
Detalhamento dos padrões de interação

Após uma análise da lista completa de Padrões de interação, foi possível identificar elementos comuns e reduzir o conjunto para uma lista consolidada.

Na Tabela A.2, apresentamos todos Padrões de interação identificados. Na coluna "Padrões de interação Propostos" temos os padrões refinados já com a intenção de simplificar o conjunto.

Também criamos uma classificação adicional para agrupar os tipos de funções de comunicação disponibilizados para a aplicação. Essas funções de suporte de comunicação são disponibilizados pela camada de comunicação da biblioteca de execução ou pelo sistema operacional. O forma de funcionamento e o fluxo de dados da comunicação depende diretamente da topologia da rede e do método de roteamento escolhido. A Tabela A.2 inclui uma coluna para identificar o "Suporte de Comunicação" utilizado pela aplicação do respectivo padrão de interação .

A seguir descrevemos os três grupos que compõem a classificação de suporte de comunicação:

1. Sem Topologia - A aplicação não enxerga a topologia da rede. A topologia de suporte à comunicação implementa o envio de mensagem de qualquer nó para a estação base. A comunicação com os vizinhos *1-hop* é disponibilizada com uma mensagem *broadcast*. Em alguns casos também é disponibilizada a comunicação com os vizinhos *n-hops*.

2. Topologia Hierárquica - A aplicação enxerga e pode fazer uso da comunicação através de uma topologia hierárquica. São disponibilizadas funções de envio de mensagens para o nó Pai e para os nós Filhos. A comunicação com os vizinhos *1-hop* é disponibilizada com uma mensagem *broadcast*. Em alguns casos também é disponibilizada a comunicação com os vizinhos *n-hops*.

3. Topologia Livre - A aplicação define a topologia através de canais de comunicação. A distribuição da rede e a capacidade de comunicação dos nós irão limitar as possibilidades de interligação dos nós. Em alguns casos pode não fazer sentido a comunicação com os vizinhos *n-hops*.

A partir da consolidação obtida, podemos resumir os padrões de interação conforme a Tabela A.1. Nessa tabela temos uma distribuição de cada padrão de interação pelos tipos de suporte de comunicação (topologias) .

Os campos marcados com `X` indicam que o respectivo padrão se aplica no domínio de uma determinado tipos de suporte de comunicação.

Observando a Tabela A.1, nota-se que ao combinarmos a coluna "ST-Sem Topologia" com a coluna "TH-Topologia Hierárquica", vamos obter um resultado que considera quase a totalidade dos padrões consolidados. A diferença final fica por conta dos padrões adicionais para implementação de um modelo de comunicação baseado em canais.

Na Tabela 3.1 do capítulo 3, apresentamos a nossa proposta final de padrões de interação para o modelo de programação proposto.

Tabela A.1: Distribuição dos Padrões de interação X Suporte de comunicação

Tipo	**Nome**	**ST**	**TH**	**ST+TH**	**TL**
Comunicação	Envia EB	X	X	X	
	Envia Nó	X		X	
	Recebe EB		X	X	
	Envia Pai		X	X	
	Envia Filhos		X	X	
	Criação de Canal				X
	Assinatura de canal				X
	Publicação no canal				X
Função Agregadora	Sumário (Grupo)	X		X	
	Reserva Recurso (Grupo)	X		X	
	Média (Grupo)	X		X	
	Média (Hierarq.)		X	X	
	Somatório (Hierarq.)		X	X	
	Máximo (Hierarq.)		X	X	
	Mínimo (Hierarq.)		X	X	
Função Local	Evento Local	X		X	
Grupo	Vizinhos *1-hop*	X	X	X	
	Vizinhos *n-hops*	X		X	X
	Vizinhos *n-hops* Condicional	X		X	
	Parâmetros dinâmicos	X	X	X	X
	Parâmetros estáticos	X	X	X	X
	Topologia hierárquica		X	X	
	Rede	X		X	X
	Distribuição proporcional				X
Suporte Agregação	Suporte Agregação	X	X	X	X

ST=Sem Topologia; TH=Topologia Hierárquica; TL=Topologia Livre

Na Tabela A.2 apresentamos a lista completa dos Padrões de interação identificados.

Tabela A.2: Padrões Identificados

Padrões identificados	Padrões propostos (#/Tipo/Nome)			Suporte de Comunicação
Abstração para comunicação				
Tipos de Agrupamentos				
Vizinhos	**1**	Grupo	Vizinhos *1-hop*	Sem Topologia
Propagação	**2**	Grupo	Vizinhos *n-hops*	Sem Topologia
Atributos estáticos	**3**	Grupo	Parâmetros estáticos	Sem Topologia
Atributos variáveis	**4**	Grupo	Parâmetros dinâmicos	Sem Topologia
Tipos de Troca de Mensagens				
Envio periódico	**5**	Comunicação	Envia EB	Sem Topologia
Envio eventual	5	Comunicação	Envia EB	Sem Topologia
Escrita individual	**6**	Comunicação	Recebe EB	Topologia Hierárquica
Escrita em grupo	6	Comunicação	Recebe EB	Topologia Hierárquica
Escrita em massa	6	Comunicação	Recebe EB	Topologia Hierárquica
Aplicações Exemplos				
Aplicação 1				
Agrupamento - Grupo de vizinhos *1-hop*	1	Grupo	Vizinhos *1-hop*	Sem Topologia
Função - Alarme local	**7**	Função Local	Evento Local	Sem Topologia
Função - Cálculo do quorum do grupo	**8**	Função Agregadora	Sumário	Sem Topologia
Comunicação - Envio de mensagem para estação base	5	Comunicação	Envia EB	Sem Topologia
Aplicação 2				
Agrupamento - Identificação do ambiente de localização do nó	3	Grupo	Parâmetros estáticos	Sem Topologia
Função - Alarme local	7	Função Local	Evento Local	Sem Topologia
Função - Cálculo do quorum do grupo	8	Função Agregadora	Sumário	Sem Topologia

Continua na próxima página...

Tabela A.2: Padrões Identificados – continuação

Padrões identificados	Padrões propostos (#/Tipo/Nome)			Função de comunicação
Função - Cálculo da média do grupo	**9**	Função Agregadora	Média	Sem Topologia
Comunicação - Envio de mensagem para estação base	5	Comunicação	Envia EB	Sem Topologia
Aplicação 3				
Agrupamento - Grupo de vizinhos *n-hops* (propagação)	2	Grupo	Vizinhos n-hops	Sem Topologia
Agrupamento - Identificação da região de localização do nó.	3	Grupo	Parâmetros estáticos	Sem Topologia
Função - Reserva de recurso num grupo	**10**	Função Agregadora	Reserva Recurso	Sem Topologia
Função - Confirmação de recurso num grupo	**11**	Comunicação	Envia Nó	Sem Topologia
Função - Cálculo dos contadores do grupo	8	Função Agregadora	Sumário	Sem Topologia
Comunicação - Envio de mensagem para estação base	5	Comunicação	Envia EB	Sem Topologia
Modelos de Macroprogramação				
Regiment				
Agrupamento - subconjunto de nós que os valores de sensores atendam determinada condição.	4	Grupo	Parâmetros dinâmicos	Sem Topologia
Agrupamento - Grupos de vizinhos *n-hops*	2	Grupo	Vizinhos *n-hops*	Sem Topologia
Agrupamento - Grupos de vizinhos distantes x metros	3	Grupo	Parâmetros estáticos	Sem Topologia
Agrupamento - Grupos de vizinhos mais próximos até k nós.	**12**	Grupo	Vizinhos *n-hops* Condicional	Sem Topologia
Comunicação - Envio de mensagem para a estação base.	5	Comunicação	Envia EB	Sem Topologia
Aplicar função f no nó local	-	Não se aplica		
Aplicar função f nos nós do grupo	-	Não se aplica		
Aplicar função f de agregação no grupo	-	Não se aplica		

Continua na próxima página...

Tabela A.2: Padrões Identificados – continuação

Padrões identificados	Padrões propostos (#/Tipo/Nome)			
Pleiades				
Agrupamento - Grupos de vizinhos *1-hop*	1	Grupo	Vizinhos *1-hop*	Sem Topologia
Agrupamento - Conjunto de todos nós da rede	**13**	Grupo	Rede	Sem Topologia
Comunicação - Nenhuma abstração disponibilizada. Deve-se utilizar o nativo do NesC/TinyOS.	-	Não se aplica		
Função wait para simular leitura síncrona de um sensor no nó local	-	Não se aplica		
Operação cfor é um loop para iteração sobre os nós de um grupo.	**14**	Suporte Agregação	Suporte Agregação	Sem Topologia
Cosmos				
Agrupamento - Agregação e difusão na hierarquia da rede	**15**	Grupo	Topologia hierárquica	Topologia Hierárquica
Agrupamento - Grupos de vizinhos *1-hop*	1	Grupo	Vizinhos *1-hop*	Topologia Hierárquica
Comunicação - Permite definir a comunicação na hierarquia dos componentes. A estação base é tratada como um componente raiz.	**16**	Comunicação	Envia Pai	Topologia Hierárquica
	17	Comunicação	Envia Filhos	Topologia Hierárquica
	5	Comunicação	Envia EB	Topologia Hierárquica
Funções - Dá suporte para agregação e difusão de mensagens, as funções devem ser desenvolvidas pelo programador em C.	14	Suporte Agregação	Suporte Agregação	Topologia Hierárquica
WADL				
Agrupamento - Agregação ou difusão por meio de filtros que identificam os componentes.	4	Grupo	Parâmetros dinâmicos	Topologia Livre
	3	Grupo	Parâmetros estáticos	Topologia Livre
Comunicação - Permite definir a comunicação na hierarquia dos componentes.	**18**	Comunicação	Publicação no canal	Topologia Livre
	19	Comunicação	Assinatura de canal	Topologia Livre
	20	Comunicação	Criação de Canal	Topologia Livre

Continua na próxima página...

Tabela A.2: Padrões Identificados – continuação

Padrões identificados	Padrões propostos (#/Tipo/Nome)			Função de comunicação
Funções - Dá suporte para agregação e difusão de mensagens, as funções devem ser desenvolvidas pelo programador em Java.	14	Suporte Agregação	Suporte Agregação	Topologia Livre
TinyDB				
Agrupamento - Agregação por meio de atributos que identificam os nós.	4	Grupo	Parâmetros dinâmicos	Topologia Hierárquica
	3	Grupo	Parâmetros estáticos	Topologia Hierárquica
Comunicação - Toda informação sempre é coletada na estação base.	5	Comunicação	Envia EB	Topologia Hierárquica
Funções - Dá suporte para agregação com algumas funções típicas de SQL.	14	Suporte Agregação	Suporte Agregação	Topologia Hierárquica
	21	Função Agregadora	Somatório	Topologia Hierárquica
	22	Função Agregadora	Média	Topologia Hierárquica
	23	Função Agregadora	Máximo	Topologia Hierárquica
	24	Função Agregadora	Mínimo	Topologia Hierárquica
ATaG				
Agrupamento - Grupos de vizinhos *n-hops*.	2	Grupo	Vizinhos *n-hops*	Topologia Livre
Agrupamento - Grupos de nós distantes d metros.	3	Grupo	Parâmetros estáticos	Topologia Livre
Agrupamento - Todos nós da rede.	13	Grupo	Rede	Topologia Livre
Agrupamento - Grupo de nós distribuídos em domínios proporcionais da rede.	**25**	Grupo	Distribuição proporcional	Topologia Livre
Comunicação - Permite definir a comunicação na hierarquia dos componentes.	18	Comunicação	Publicação no canal	Topologia Livre
	19	Comunicação	Assinatura de canal	Topologia Livre
	20	Comunicação	Criação de Canal	Topologia Livre
Funções - Dá suporte para agregação e difusão de mensagens, as funções devem ser desenvolvidas pelo programador em Java.	14	Suporte Agregação	Suporte Agregação	Topologia Livre

B
Diagramas das FSMs e Parâmetros de Configuração

B.1
Aplicação utilizada nos testes

Tabela B.1: Configuração da primeira FSM do primeiro teste

Id	Current State	Parent Id	Parent State	Event Param	Action	New State
Startup						
7	Begin	7	Begin	booted_S	init()	Initializing
7	Initializing	7	Initializing	initDone_S	CollInit()	Initiating Collections
7	Initiating Collections	7	Initiating Collections	CollInitDone_S	dummy()	Idle
7	Idle	7	Idle	TimerFired_S	nextColl()	Idle
7	Idle	7	Idle	startCollection_S	dummy()	Collecting
7	Collecting	7	Collecting	TimerFired_S	lostColl()	Collecting
7	Collecting	7	Collecting	endCollection_S	nextColl()	Idle

Tabela B.2: Configuração da segunda FSM do primeiro teste

Id	Current State	Parent Id	Parent State	Event Param	Action	New State
Monitor - Agrega e envia para EB						
0	Begin	0	Begin	CollInitDone_S	dummy()	Waiting Start
0	Waiting Start	0	Waiting Start	nextColl_S	testCoord()	Testing Coord
0	Testing Coord	0	Testing Coord	testCoordDone_T	startAggreg()	Aggregating
0	Testing Coord	0	Testing Coord	testCoordDone_F	endCollection()	Waiting Start
0	Aggregating	0	Aggregating	AggregDone_T	sendBSAggValue()	Sending BS
0	Aggregating	0	Aggregating	AggregDone_F	sendBSAggValue()	Sending BS
0	Sending BS	0	Sending BS	sendBSDone_S	endCollection()	Waiting Start
0	Sending BS	0	Sending BS	sendBSDone_E	endCollection()	Waiting Start

Tabela B.3: Configuração da terceira FSM utilizada no segundo teste

Id	Current State	Parent Id	Parent State	Event Param	Action	New State
Alarm - Lê valor local, se a comparação for T, agrega e sempre envia para EB						
1	Begin	1	Begin	CollInitDone_S	dummy()	Waiting Start
1	Waiting Start	7	Collecting	nextColl_S	readSensor()	Reading Sensor
1	Reading Sensor	1	Reading Sensor	readSensorDone_S	testTrigger()	Testing Trigger
1	Testing Trigger	1	Testing Trigger	testTriggerDone_F	endCollection()	Waiting Start
1	Testing Trigger	1	Testing Trigger	testTriggerDone_T	startAggreg()	Sending BS
1	Sending BS	7	Collecting	sendBSDone_S	endCollection()	waiting Start
1	Sending BS	7	Collecting	sendBSDone_E	endCollection()	waiting Start

B.2
Aplicações de Referência

B.2.1
Diagramas das FSMs

Aplicação 1

Apresentamos os diagramas das máquinas de estado da aplicação 1 nas figuras B.1 e B.2.

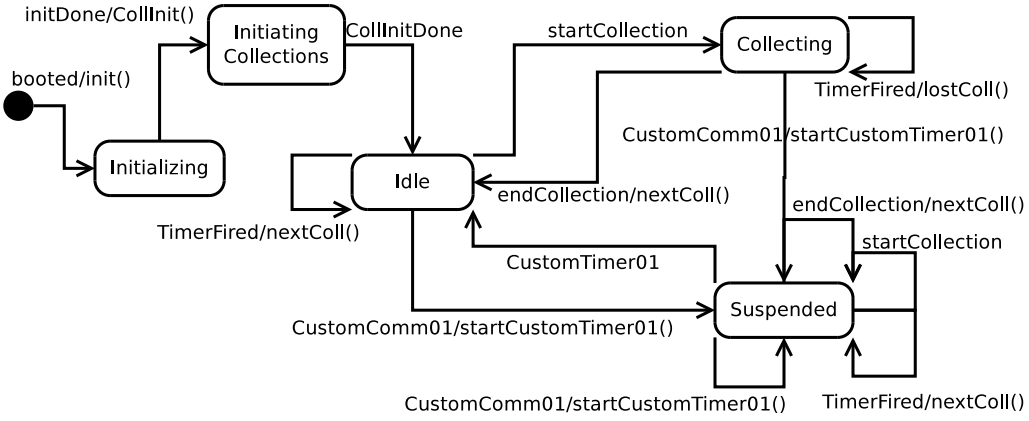

Figura B.1: Aplicação 1 - FSM para Inicialização do Mote e controle geral

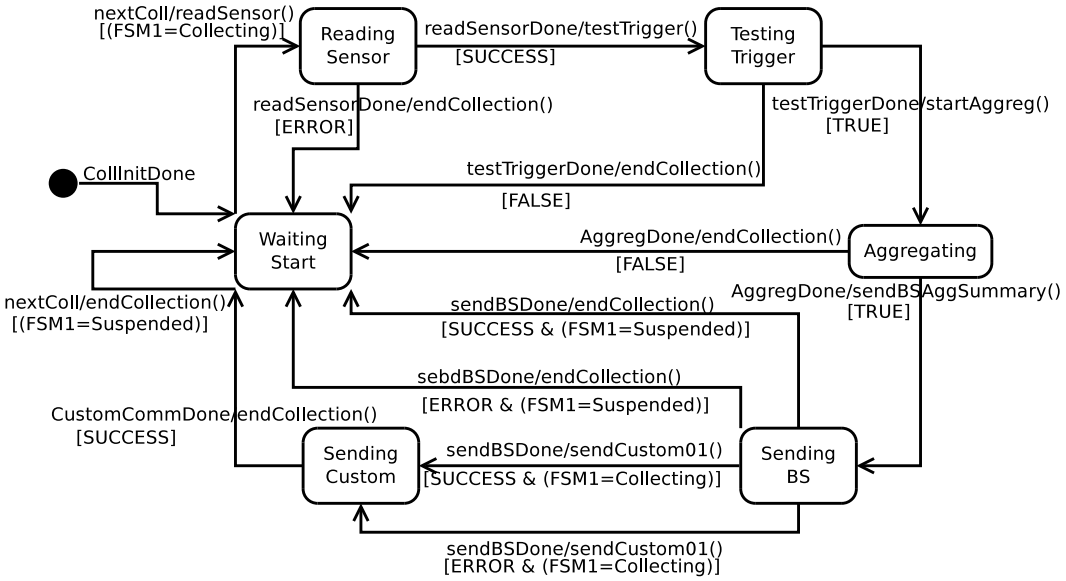

Figura B.2: Aplicação 1 - FSM para geração do alarme

Aplicação 2

Apresentamos os diagramas das máquinas de estado da aplicação 2 nas figuras B.3, B.4 e B.5.

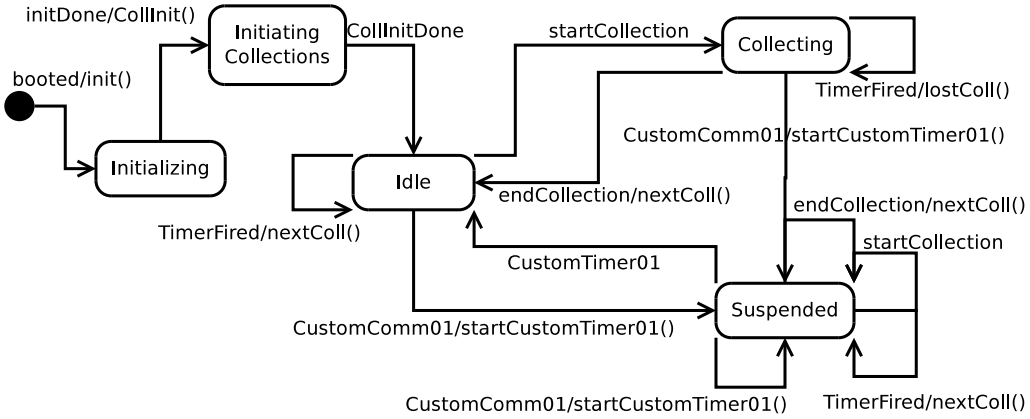

Figura B.3: Aplicação 2 - FSM para Inicialização do Mote e controle geral

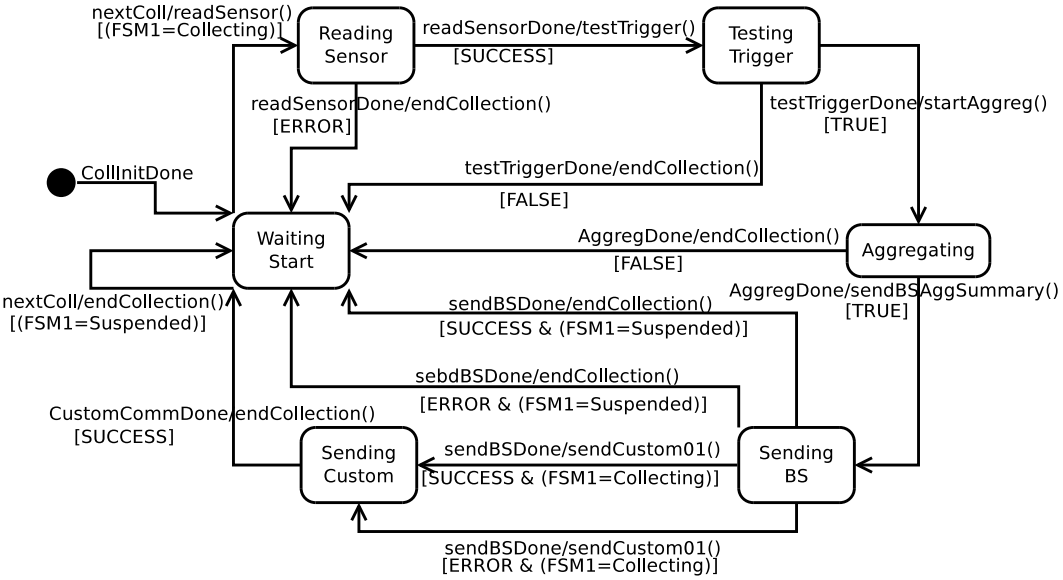

Figura B.4: Aplicação 2 - FSM para geração do alarme

Aplicação 3

Apresentamos os diagramas das máquinas de estado da aplicação 3 nas figuras B.6, B.7 e B.8.

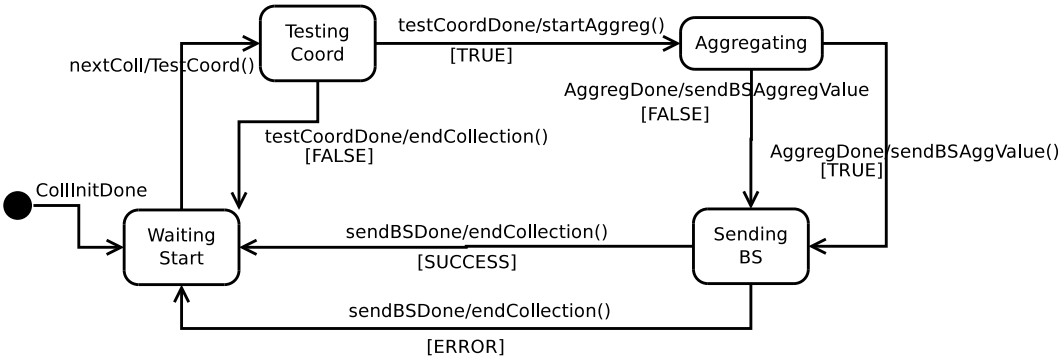

Figura B.5: Aplicação 2 - FSM para monitoração periódica

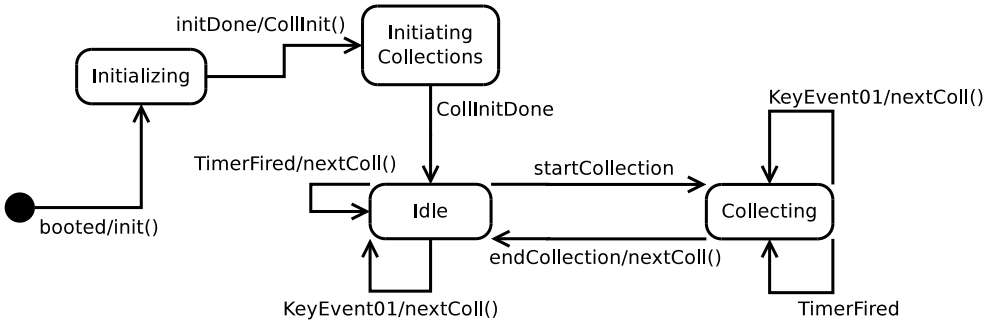

Figura B.6: Aplicação 3 - FSM para Inicialização do Mote e controle geral

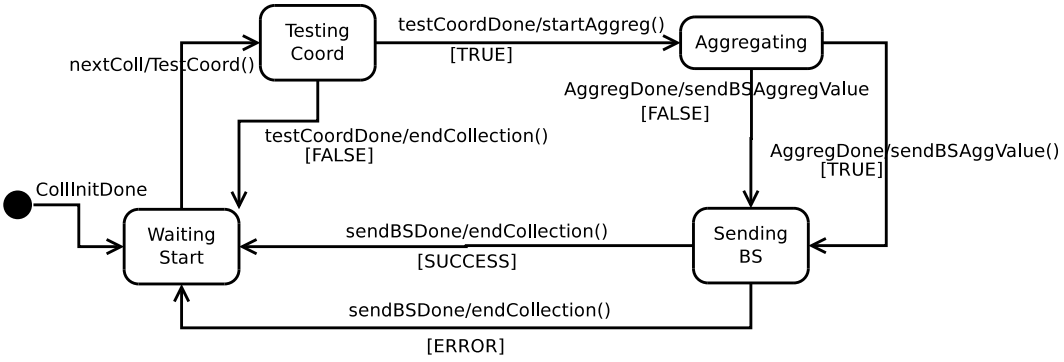

Figura B.7: Aplicação 3 - FSM para monitoração periódica

B.2.2
Parâmetros de configuração

Apresentamos nas Tabelas B.4, B.5 e B.6, as configurações dos parâmetros para as três aplicações de referência.

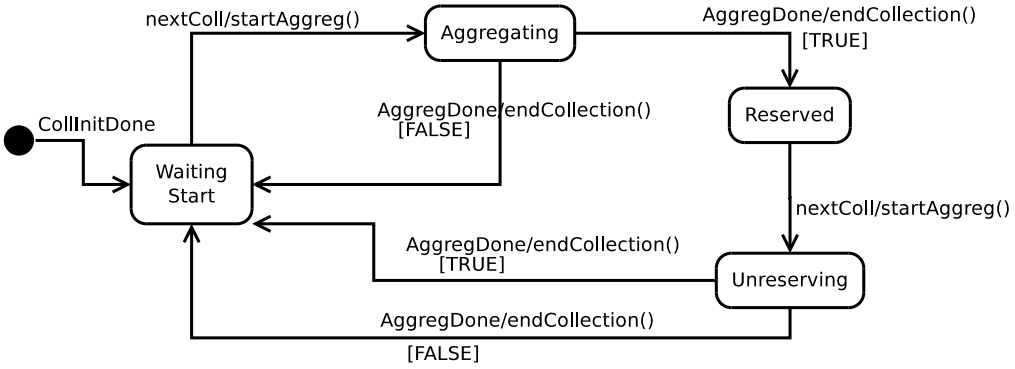

Figura B.8: Aplicação 3 - FSM para reserva de vagas

Tabela B.4: Estrutura dos parâmetros de configuração para Aplicação 1

Campo	**Value**	**Descrição**
Parâmetros Gerais		
CTimer01Delay	70	Custom Timer 1 Delay (segundos)
ElectionPeriod	60	Período para reeleição de coordenador (minutos)
Configuração da Coleta 1 - Alarme		
GroupDef	0x0101	Definição da formação do grupo
MaxHops	1	Max hops do grupo
TimerPeriod	30000	Período da coleta do grupo
LocalStage InputCompOper	0x03	Operação Local: Identifica o sensor e a operação de comparação
LocalStage RefValue	29	Operação Local: Valor de referência para comparação
AggregOper	4	Define a operação de agregação utilizada
AggregStage1 InputCompOper	0x03	Agregação Estágio 1: Identifica a entrada e a operação de comparação
AggregStage1 RefValue	26	Agregação Estágio 1: Valor de referência para comparação
AggregStage2 InputCompOper	0x03	Agregação Estágio 2: Identifica a entrada e a operação de comparação
AggregStage2 RefValue	2	Agregação Estágio 2: Valor de referência para comparação

Tabela B.5: Estrutura dos parâmetros de configuração para Aplicação 2

Campo	**Value**	**Descrição**
Parâmetros Gerais		
GroupA	10	Parâmetro A utilizados na definição de grupos
CTimer01Delay	70	Custom Timer 1 Delay (segundos)
ElectionPeriod	60	Período para reeleição de coordenador (minutos)
Configuração da Coleta 1		
GroupDef	0x0101	Definição da formação do grupo
MaxHops	3	Max hops do grupo
TimerPeriod	30000	Período da coleta do grupo
LocalStage InputCompOper	0x03	Operação Local: Identifica o sensor e a operação de comparação
LocalStage RefValue	29	Operação Local: Valor de referência para comparação
AggregOper	4	Define a operação de agregação utilizada
AggregStage1 InputCompOper	0x03	Agregação Estágio 1: Identifica a entrada e a operação de comparação
AggregStage1 RefValue	29	Agregação Estágio 1: Valor de referência para comparação
AggregStage2 InputCompOper	0x04	Agregação Estágio 2: Identifica a entrada e a operação de comparação
AggregStage2 RefValue	0	Agregação Estágio 2: Valor de referência para comparação
Configuração da Coleta 2		
GroupDef	0x0109	Definição da formação do grupo
MaxHops	3	Max hops do grupo
TimerPeriod	1000	Período da coleta do grupo
AggregOper	0	Define a operação de agregação utilizada

Tabela B.6: Estrutura dos parâmetros de configuração para Aplicação 3

Campo	Value	Descrição
Parâmetros Gerais		
GroupA	10	Parâmetro A utilizados na definição de grupos
ElectionPeriod	60	Período para reeleição de coordenador (minutos)
Configuração da Coleta 1		
GroupDef	0x0109	Definição da formação do grupo
MaxHops	7	Max hops do grupo
TimerPeriod	30000	Período da coleta do grupo
LocalStage InputCompOper	0x00	Operação Local: Identifica o sensor e a operação de comparação
LocalStage RefValue	0	Operação Local: Valor de referência para comparação
AggregOper	4	Define a operação de agregação utilizada
AggregStage1 InputCompOper	0x11	Agregação Estágio 1: Identifica a entrada e a operação de comparação
AggregStage1 RefValue	1	Agregação Estágio 1: Valor de referência para comparação
AggregStage2 InputCompOper	0x04	Agregação Estágio 2: Identifica a entrada e a operação de comparação
AggregStage2 RefValue	0	Agregação Estágio 2: Valor de referência para comparação
Configuração da Coleta 2		
GroupDef	0x0001	Definição da formação do grupo
MaxHops	3	Max hops do grupo
TimerPeriod	0	Período da coleta do grupo
LocalStage InputCompOper	0x00	Operação Local: Identifica o sensor e a operação de comparação
LocalStage RefValue	0	Operação Local: Valor de referência para comparação
AggregOper	5	Define a operação de agregação utilizada
AggregStage1 InputCompOper	0x11	Agregação Estágio 1: Identifica a entrada e a operação de comparação
AggregStage1 RefValue	0	Agregação Estágio 1: Valor de referência para comparação
AggregStage2 InputCompOper	0x00	Agregação Estágio 2: Identifica a entrada e a operação de comparação
AggregStage2 RefValue	0	Agregação Estágio 2: Valor de referência para comparação

B.2.3 Tabelas FSM

Apresentamos nas Tabelas B.7, B.8, B.9, B.10, B.11, B.12, B.13, as transições das FSMs para as três aplicações de referência.

Tabela B.7: Aplicação 1 - Máquina de estados 1 (Inicialização)

Id	Current State	Parent Id	Parent State	Event Param	Action	New State
Startup and General control						
7	Begin	7	Begin	booted_S	init()	Initializing
7	Initializing	7	Initializing	initDone_S	CollInit()	Initiating Collections
7	Initiating Collections	7	Initiating Collections	CollInitDone_S	dummy()	Idle
7	Idle	7	Idle	TimerFired_S	nextColl()	Idle
7	Idle	7	Idle	startCollection_S	dummy()	Collecting
7	Idle	7	Idle	CustomComm01_S	startCustomTimer01()	Suspended
7	Collecting	7	Collecting	TimerFired_S	lostColl()	Collecting
7	Collecting	7	Collecting	endCollection_S	nextColl()	Idle
7	Collecting	7	Collecting	CustomComm01_S	startCustomTimer01()	Suspended
7	Suspended	7	Suspended	CustomTimer01_S	dummy()	Idle
7	Suspended	7	Suspended	TimerFired_S	nextColl()	Suspended
7	Suspended	7	Suspended	CustomComm01_S	startCustomTimer01()	Suspended
7	Suspended	7	Suspended	endCollection_S	nextColl()	Suspended
7	Suspended	7	Suspended	startCollection_S	dummy()	Suspended

Tabela B.8: Aplicação 1 - Máquina de estados 2 (Alarme)

Id	Current State	Parent Id	Parent State	Event Param	Action	New State
Alarm - Lê valor local, se a comparação for T, agrega e sempre envia para EB						
0	Begin	0	Begin	CollInitDone_S	dummy()	Waiting Start
0	Waiting Start	7	Collecting	nextColl_S	readSensor()	Reading Sensor
0	Waiting Start	7	Suspended	nextColl_S	endCollection()	Waiting Start
0	Reading Sensor	0	Reading Sensor	readSensorDone_S	testTrigger()	Testing Trigger
0	Reading Sensor	0	Testing Trigger	readSensorDone_E	endCollection()	Waiting Start
0	Testing Trigger	0	Testing Trigger	testTriggerDone_F	endCollection()	Waiting Start
0	Testing Trigger	0	Testing Trigger	testTriggerDone_T	startAggreg()	Aggregating
0	Aggregating	0	Aggregating	AggregDone_T	sendBSAggSummary()	Sending BS
0	Aggregating	0	Aggregating	AggregDone_F	endCollection()	Waiting Start
0	Sending BS	7	Collecting	sendBSDone_S	sendCustom01()	Sendind Custom
0	Sending BS	7	Suspended	sendBSDone_S	endCollection()	waiting Start
0	Sending BS	7	Collecting	sendBSDone_E	sendCustom01()	Sendind Custom
0	Sending BS	7	Suspended	sendBSDone_E	endCollection()	Waiting Start
0	Sendind Custom	0	Sendind Custom	CustomCommDone_S	endCollection()	Waiting Start

Tabela B.9: Aplicação 2 - Máquina de estados 1 (Inicialização)

Id	Current State	Parent Id	Parent State	Event Param	Action	New State
Startup and General control						
7	Begin	7	Begin	booted_S	init()	Initializing
7	Initializing	7	Initializing	initDone_S	CollInit()	Initiating Collections
7	Initiating Collections	7	Initiating Collections	CollInitDone_S	dummy()	Idle
7	Idle	7	Idle	TimerFired_S	nextColl()	Idle
7	Idle	7	Idle	startCollection_S	dummy()	Collecting
7	Idle	7	Idle	CustomComm01_S	startCustomTimer01()	Suspended
7	Collecting	7	Collecting	TimerFired_S	lostColl()	Collecting
7	Collecting	7	Collecting	endCollection_S	nextColl()	Idle
7	Collecting	7	Collecting	CustomComm01_S	startCustomTimer01()	Suspended
7	Suspended	7	Suspended	CustomTimer01_S	dummy()	Idle
7	Suspended	7	Suspended	TimerFired_S	nextColl()	Suspended
7	Suspended	7	Suspended	CustomComm01_S	startCustomTimer01()	Suspended
7	Suspended	7	Suspended	endCollection_S	nextColl()	Suspended
7	Suspended	7	Suspended	startCollection_S	dummy()	Suspended

Tabela B.10: Aplicação 2 - Máquina de estados 2 (Alarme)

Id	Current State	Parent Id	Parent State	Event Param	Action	New State
Alarm - Lê valor local, se a comparação for T, agrega, se resultado for verdadeiro envia para EB						
0	Begin	0	Begin	CollInitDone_S	dummy()	Waiting Start
0	Waiting Start	7	Collecting	nextColl_S	readSensor()	Reading Sensor
0	Waiting Start	7	Suspended	nextColl_S	endCollection()	Waiting Start
0	Reading Sensor	0	Reading Sensor	readSensorDone_S	testTrigger()	Testing Trigger
0	Reading Sensor	0	Reading Sensor	readSensorDone_E	endCollection()	Waiting Start
0	Testing Trigger	0	Testing Trigger	testTriggerDone_F	endCollection()	Waiting Start
0	Testing Trigger	0	Testing Trigger	testTriggerDone_T	startAggreg()	Aggregating
0	Aggregating	0	Aggregating	AggregDone_T	sendBSAggSummary()	Sending BS
0	Aggregating	0	Aggregating	AggregDone_F	endCollection()	Waiting Start
0	Sending BS	7	Collecting	sendBSDone_S	sendCustom01()	Sendind Custom
0	Sending BS	7	Suspended	sendBSDone_S	endCollection()	waiting Start
0	Sending BS	7	Collecting	sendBSDone_E	sendCustom01()	Sendind Custom
0	Sending BS	7	Suspended	sendBSDone_E	endCollection()	Waiting Start
0	Sendind Custom	0	Sendind Custom	CustomCommDone_S	endCollection()	Waiting Start

Tabela B.11: Aplicação 2 - Máquina de estados 3 (Monitoração)

Id	Current State	Parent Id	Parent State	Event Param	Action	New State
Monitor - Agrega e envia para EB						
1	Begin	1	Begin	CollInitDone_S	dummy()	Waiting Start
1	Waiting Start	1	Waiting Start	nextColl_S	testCoord()	Testing Coord
1	Testing Coord	1	Testing Coord	testCoordDone_T	startAggreg()	Aggregating
1	Testing Coord	1	Testing Coord	testCoordDone_F	endCollection()	Waiting Start
1	Aggregating	1	Aggregating	AggregDone_T	sendBSAggValue()	Sending BS
1	Aggregating	1	Aggregating	AggregDone_F	sendBSAggValue()	Sending BS
1	Sending BS	1	Sending BS	sendBSDone_S	endCollection()	Waiting Start
1	Sending BS	1	Sending BS	sendBSDone_E	endCollection()	Waiting Start

Tabela B.12: Aplicação 3 - Máquina de estados 1 (Inicialização)

Id	Current State	Parent Id	Parent State	Event Param	Action	New State
Startup and General control						
7	Begin	7	Begin	booted_S	init()	Initializing
7	Initializing	7	Initializing	initDone_S	CollInit()	Initiating Collections
7	Initiating Collections	7	Initiating Collections	CollInitDone_S	dummy()	Idle
7	Idle	7	Idle	TimerFired_S	nextColl()	Idle
7	Idle	7	Idle	KeyEvent01_S	nextColl()	Idle
7	Idle	7	Idle	startCollection_S	dummy()	Collecting
7	Collecting	7	Collecting	TimerFired_S	dummy()	Collecting
7	Collecting	7	Collecting	KeyEvent01_S	nextColl()	Collecting
7	Collecting	7	Collecting	endCollection_S	nextColl()	Idle

Tabela B.13: Aplicação 3 - Máquina de estados 2 (Monitoração) e 3 (Reserva)

Id	Current State	Parent Id	Parent State	Event Param	Action	New State
Monitor - Agrega e envia para EB						
0	Begin	0	Begin	CollInitDone_S	dummy()	Waiting Start
0	Waiting Start	7	Collecting	nextColl_S	testCoord()	Testing Coord
0	Testing Coord	0	Testing Coord	testCoordDone_T	startAggreg()	Aggregating
0	Testing Coord	0	Testing Coord	testCoordDone_F	endCollection()	Waiting Start
0	Aggregating	0	Aggregating	AggregDone_T	sendBSAggValue()	Sending BS
0	Aggregating	0	Aggregating	AggregDone_F	sendBSAggValue()	Sending BS
0	Sending BS	0	Sending BS	sendBSDone_S	endCollection()	Waiting Start
0	Sending BS	0	Sending BS	sendBSDone_E	endCollection()	Waiting Start
Reserve - Evento manual para Agregação						
1	Begin	1	Begin	CollInitDone_S	dummy()	Waiting Start
1	Waiting Start	1	Waiting Start	nextColl_S	startAggreg()	Aggregating
1	Aggregating	1	Aggregating	AggregDone_T	endCollection()	Reserved
1	Aggregating	1	Aggregating	AggregDone_F	endCollection()	Waiting Start
1	Reserved	1	Reserved	nextColl_S	startAggreg()	Unreserving
1	Unreserving	1	Unreserving	AggregDone_T	endCollection()	Waiting Start
1	Unreserving	1	Unreserving	AggregDone_F	endCollection()	Waiting Start

C
Ambiente de execução para os testes

Os testes mais exaustivos e monitorados foram executados num ambiente simulado baseado na ferramenta TOSSIM[14]. Para isso foi necessário construir, além do nosso sistema e do módulo de controle em JAVA, alguns módulos auxiliares para os testes. Na Figura C.1, temos um diagrama com todos os módulos envolvidos nos testes.

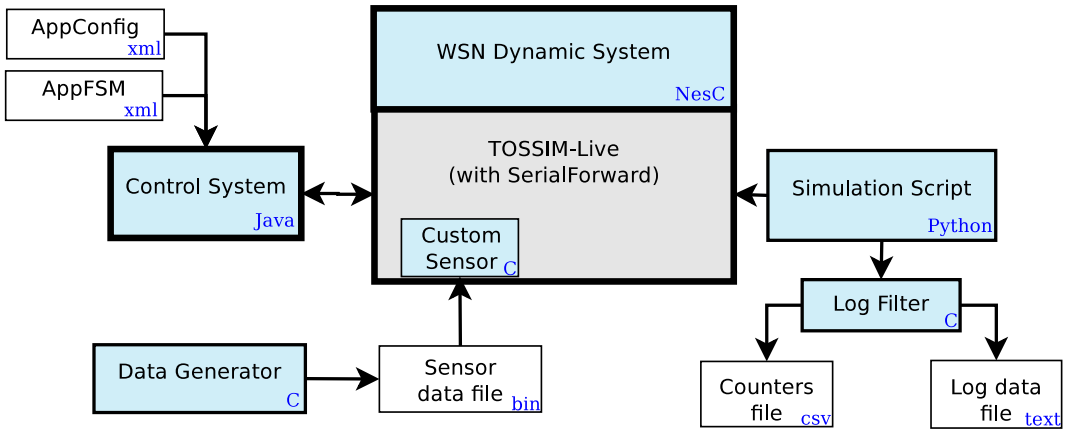

Figura C.1: Módulos utilizados para execução dos testes no simulador

A seguir, descrevemos a função de cada módulo.

TOSSIM-Live - Faz parte do pacote do TinyOS e é o principal suporte para simulação da aplicação. Simula o funcionamento de uma rede de motes MICAz a partir de um script em Python.

WSN Dynamic System - É o nosso sistema que implementa o modelo de programação proposto.

Simulation Script - Script em Python para o funcionamento da rede, define os nós ativos e o alcance de rádio entre eles. Também captura as mensagens de log da aplicação e dos componentes do simulador.

Log Filter - Módulo responsável para extrair das mensagens de logs os contadores de envio de mensagens e substituir os códigos numéricos da FSM pelos respectivos textos.

Control System - Sistema em Java que funciona como módulo servidor da aplicação. Responsável pela configuração dos motes e pelo recebimento dos dados. Comunica-se com o simulador através da interface Serial-Forward.

Data Generator - Módulo para geração do arquivo de dados simulados para os sensores.

Custom Sensor - Componente inserido no simulador para permitir a leitura do arquivo de dados dos sensores.

D
Interfaces e Componentes NesC do sistema

Tabela D.1: Interfaces das Camadas Funcionais

Tipo	**Nome**	**Descrição**
Interface: AggregMainExt		
Cmd	init()	Inicializa a camada
Evt	initDone()	Sinaliza o final da inicialização
Cmd	start()	Dispara uma nova agregação
Evt	done()	Sinaliza o final de uma agregação
Interface: GroupMainExt		
Cmd	init()	Inicializa a camada
Evt	initDone()	Sinaliza o final da inicialização
Cmd	RequestValues()	Dispara uma coleta de grupo
Evt	valueReturn()	Sinaliza cada valor retornado
Interface: GroupMainAux		
Cmd	sendCustomComm()	Envia um comando customizado para o grupo
Interface: CommMainExt		
Cmd	init()	Inicializa a camada
Event	initDone()	Sinaliza o final da inicialização
Cmd	sendNHops()	Dispara uma mensagem para propagação no grupo
Cmd	sendNhopsReturn()	Retorna uma mensagem com valor solicitado
Cmd	nextMessage()	Libera a leitura da próxima mensagem recebida
Evt	recNhopsReturn()	Recebe uma mensagem com valor solicitado
Evt	recNhops()	Recebe uma mensagem de propagação
Interface: Communication		
Cmd	initComm()	Inicializa a camada
Event	initCommDone()	Sinaliza o final da inicialização

Continua na próxima página...

Tabela D.1: Interfaces das Camadas Funcionais – continuação

Tipo	**Nome**	**Descrição**
Cmd	loadFSMTable()	Carrega uma nova tabela FSM (somente Base Station)
Evt	newFSMTable()	Sinaliza a carga de uma nova tabela FSM
Cmd	sendNHops()	Envia uma mensagem NHops
Cmd	sendNHopsReturn()	Envia uma mensagem NHopsReturn
Cmd	sendDataBS()	Envia uma mensagem de dados para BS
Cmd	sendReqConf()	Envia uma mensagem ReqConf para a BS
Evt	recNHops()	Sinaliza o recebimento de uma mensagem NHops
Evt	recNHopsReturn()	Sinaliza o recebimento de uma mensagem NHopsReturn
Evt	recMoteConfA()	Sinaliza o recebimento de uma mensagem MoteConfA
Evt	recMoteConfB()	Sinaliza o recebimento de uma mensagem MoteConfB
Cmd	nextMessage()	Libera a leitura da próxima mensagem recebida
Interface: CommunicationAux		
Cmd	sendBS()	Envia uma mensagem de dados para BS
Evt	sendBSdone()	Sinaliza o envio de uma mensagem para a BS
Evt	newFSM()	Sinaliza que uma nova tabela FSM foi carregada
Evt	invalidateFSM()	Sinaliza que a tabela FSM está inválida
Cmd	getFSMAddr()	Retorna o endereço da estrutura de dados da tabela FSM

Tabela D.2: Novos componentes NesC

Configuração	**Módulo**	**Interface**
AggregMainC.nc	AggregMainP.nc	AggregMainExt.nc
AppMainAppC.nc	AppMainC.nc	
BaseStationC.nc	BaseStationP.nc	BaseStation.nc
CommMainC.nc	CommMainP.nc	CommMainExt.nc
CommunicationC.nc	CommunicationP.nc	Communication.nc, CommunicationAux.nc
dataQueueC.nc	dataQueueP.nc	dataQueue.nc
FSMC.nc	FSMP.nc	FSM.nc
	FSMTable_AggregC.nc, FSMTable_AppC.nc, FSMTable_CommC.nc, FSMTable_GroupC.nc	FSMTable.nc
GroupMainC.nc	GroupMainP.nc	GroupMainExt.nc. GroupMainAux.nc
LocalNodeC.nc	LocalNodeP.nc	LocalNode.nc, LocalNodeKey.nc
	MoteDataC.nc	MoteData.nc, MoteDataLoad.nc, MoteDataOper.nc
RTimerC.nc, RTimerP.nc	VTimerC.nc	VTimer.nc
RTimerSecC.nc, RTimerSecP.nc	VTimerSecC.nc	VTimerSec.nc
WSNDynBSAppC.nc	WSNDynBSC.nc	

E
Formatação dos arquivos de configuração XML

Formato para o arquivo de Parâmetros

```
<?xml version="1.0"?>
<AppConfig>
<Header>
 <StaticA>10</StaticA>
 <StaticB>20</StaticB>
 <StaticC>30</StaticC>
 <Ctimer01Delay>70</Ctimer01Delay>
 <Ctimer02Delay>70</Ctimer02Delay>
 <Ctimer03Delay>70</Ctimer03Delay>
 <Ctimer04Delay>70</Ctimer04Delay>
 <ElectionPeriod>8</ElectionPeriod>
</Header>
<CollConfig0>
 <GroupDef>0x0101</GroupDef>
 <MaxHops>10</MaxHops>
 <TimerPeriod>30000</TimerPeriod>
 <LocalStage_Input_CompOper>0x03</LocalStage_Input_CompOper>
 <LocalStage_RefValue>29</LocalStage_RefValue>
 <AggregOper>4</AggregOper>
 <AggregStage1_Input_CompOper>0x03</AggregStage1_Input_CompOper>
 <AggregStage1_RefValue>26</AggregStage1_RefValue>
 <AggregStage2_Input_CompOper>0x03</AggregStage2_Input_CompOper>
 <AggregStage2_RefValue>2</AggregStage2_RefValue>
</CollConfig0>
<CollConfig1>
 <GroupDef>0x0000</GroupDef>
 <MaxHops>0</MaxHops>
 <TimerPeriod>0</TimerPeriod>
 <LocalStage_Input_CompOper>0x00</LocalStage_Input_CompOper>
 <LocalStage_RefValue>0</LocalStage_RefValue>
```

```
 <AggregOper>0</AggregOper>
 <AggregStage1_Input_CompOper>0x00</AggregStage1_Input_CompOper>
 <AggregStage1_RefValue>0</AggregStage1_RefValue>
 <AggregStage2_Input_CompOper>0x00</AggregStage2_Input_CompOper>
 <AggregStage2_RefValue>0</AggregStage2_RefValue>
</CollConfig1>
<CollConfig2>
 <GroupDef>0</GroupDef>
 <MaxHops>0</MaxHops>
 <TimerPeriod>0</TimerPeriod>
 <LocalStage_Input_CompOper>0x00</LocalStage_Input_CompOper>
 <LocalStage_RefValue>0</LocalStage_RefValue>
 <AggregOper>0</AggregOper>
 <AggregStage1_Input_CompOper>0x00</AggregStage1_Input_CompOper>
 <AggregStage1_RefValue>0</AggregStage1_RefValue>
 <AggregStage2_Input_CompOper>0x00</AggregStage2_Input_CompOper>
 <AggregStage2_RefValue>0</AggregStage2_RefValue>
</CollConfig2>
</AppConfig>
```

Formato para o arquivo com a tabela FSM

A seguir representamos a formatação para somente um registro de transição. Uma configuração normal pode conter até 40 registros de transições.

```
<?xml version="1.0"?>
<TranList>
<TranReg>
  <MachineID>9</MachineID>
  <CurrState>1</CurrState>
  <ParentMachine>9</ParentMachine>
  <ParentState>1</ParentState>
  <Event>1</Event>
  <Param>0</Param>
  <Action>1</Action>
  <NewState>2</NewState>
  </TranReg>
</TranList>
```