Предметно-ориентированный язык моделирования приложений

для беспроводных сенсорных сетей

Статья посвящена использованию предметно-ориентированного подхода для моделирования приложений беспроводных сенсорных сетей (БСС). Авторами предлагается предметно-ориентированный язык (DSL), разработанный специально для создания управляющих приложений БСС.

Наталия Клименко
tinddae@mail.ru

Максим Сергиевский
sermax@yandex.ru

Беспроводные сенсорные сети, по мнению аналитиков Gartner, считаются (по крайней мере, на ближайшие 10 лет) одной из наиболее перспективных технологий. В настоящее время БСС этот прогноз вполне оправдывают, являясь динамично развивающейся прикладной областью, финансовые вложения в которую ежегодно возрастают приблизительно на порядок.

В то же время наиболее актуальным подходом к созданию программного обеспечения в последнее время становится разработка на основе моделей. Этот метод в идеале должен позволить разработчику создавать программное обеспечение, описывая его функции на некотором высокоуровневом языке в виде набора моделей функционирования, а затем автоматически переводить их в программный код.

В данной статье речь пойдет о соединении этих двух актуальных технологий, точнее — о применении подхода разработки на основе моделей в области беспроводных сенсорных сетей. Авторами статьи предлагается многоуровневый язык описания функционирования беспроводных сенсорных сетей, который после реализации в среде платформы Microsoft SQL Server Modeling можно применять для создания управляющих приложений.

Прежде всего обсудим особенности БСС, затем возможности платформы SQL Server Modeling и, наконец, рассмотрим предлагаемый язык.

Разработка приложений для БСС

Беспроводная сенсорная сеть представляет собой совокупность миниатюрных вычислительных устройств, снабженных датчиками и способных к передаче данных по радиоканалам. Эти устройства называют «мотами» (от английского *mote* — «пылинка»). Важным элементом сети является базовая станция (или шлюз), на которую поступает вся собираемая датчиками информация. На базовой станции сенсорная информация проходит предварительную обработку и передается далее в корпоративную сеть для дальнейшего анализа и использования.

Моты, образующие сеть, связаны между собой беспроводными радиоканалами. Выбор маршрутов коммуникации осуществляется динамически по алгоритмам, реализуемым протоколами связи. Передача сообщений по сети происходит поэтапно, от одного мота другому. Достаточно, чтобы от каждого мота существовал хотя бы один путь к базовой станции, однако на практике таких путей строится множество, что обеспечивает отказоустойчивость сети и увеличивает скорость передачи сообщений.

Далее рассмотрим важные особенности БСС, отличающие их как от проводных сенсорных, так и от обыкновенных беспроводных сетей.

Развертывание БСС осуществляется предельно просто, поскольку моты являются автономными устройствами и прокладки кабелей не требуется. Моты могут быть помещены в заранее определенные места или распределены случайным образом. Способность узлов БСС к самоорганизации гарантирует создание сети в случае выполнения единственного условия: связности. При этом количество узлов в сети может достигать нескольких тысяч.

Механизм самоорганизации, делающий таким простым развертывание БСС, также обеспечивает ее самовосстановление. Если какой-либо из узлов выходит из строя, сеть автоматически перенастраивается, чтобы компенсировать его отсутствие. Многие алгоритмы передачи данных в БСС также позволяют обходить участки сети с помехами и участки с низким уровнем заряда батарей, передавая сообщения по более выгодным маршрутам. Однако БСС имеют ряд принципиальных ограничений:

малая пропускная способность беспроводного канала, чувствительность к помехам, сложность организации коммуникаций при высокой плотности узлов, небольшой запас энергии и невысокая вычислительная мощность мотов.

Ширина диапазона радиопередачи ограничена, а одновременное вещание двух и более устройств на одной частоте ведет к необратимому искажению сигнала и, следовательно, к потере всех одновременно передаваемых сообщений. Соответственно, передача данных в сети должна быть организована таким образом, чтобы никакие два устройства не работали одновременно на одной частоте в радиусе действия третьего устройства. Это предъявляет жесткие требования к плотности размещения узлов. Слишком плотное размещение мотов приводит к появлению помех. Решить проблему можно уменьшением радиуса действия узлов с помощью специальной настройки приемопередатчиков. Однако помехи, присутствующие во внешней среде, к которым БСС так же чувствительны, как и к внутренним помехам, остаются слабым местом данной технологии.

Ограниченная вычислительная мощность мотов требует использования в них простых алгоритмов и не позволяет производить объемные вычисления. Моты не способны поддерживать сложные сетевые протоколы, обеспечивающие оптимальную маршрутизацию.

Стремление максимально продлить срок службы сети ведет к требованию экономии энергии батарей. Потребителями энергии в БСС являются сенсор (датчик и аналогово-цифровой преобразователь), приемопередатчик сообщений и сам вычислитель. Одной из важнейших мер экономии энергии является использование для узлов «спящего» режима, когда отдельные устройства мота отключаются по определенной временной схеме. Другим способом экономии энергии является выбор оптимального пути передачи сообщений, причем оптимальность в каждом конкретном случае трактуется поразному. К примеру, оптимальным может быть путь с минимальными помехами, что позволяет не тратить энергию на повторную пересылку потерянных сообщений, или путь, проходящий через узлы, сохранившие максимальный запас энергии. Кроме того, протоколы передачи данных должны минимизировать число передач дублирующих друг друга сообщений. При функционировании БСС велика вероятность, что данные, собранные близко расположенными сенсорами, дублируются. Поэтому желательно, предваряя пересылку, выполнять агрегирование данных.

Однако слишком строгий режим экономии, например длительное пребывание мотов в «спящем» режиме, передача сообщений без подтверждения доставки или агрегирование с утратой значащей информации, приводит к уменьшению скорости реакции сети, к снижению точности и даже потере данных. Таким образом, выделяются следующие основные характеристики БСС: надежность и скорость передачи данных, длительность работы, качество мониторинга и безопасность.

Различные приложения предъявляют принципиально разные требования к конкретным реализациям БСС. В настоящее время предложены несколько типов архитектуры сетей и огромное количество различных алгоритмов, описывающих их функционирование.

На основе проведенных исследований можно выделить основные особенности БСС, учет которых необходим при создании программных приложений:

- Архитектура сенсорных узлов. Моты состоят из нескольких компонентов, в число которых обязательно входят процессор и передатчик; кроме того, обычно присутствуют датчики, память, батарея. Таким образом, состав компонентов мота может различаться для разных устройств даже в рамках одной сети (гетерогенная сеть).
- Работа в реальном времени. В большинстве приложений беспроводные сенсорные сети работают в режиме реального времени, то есть передают информацию по мере ее поступления, отвечают на посылаемые запросы, обмениваются технологическими сообщениями.
- Обмен сообщениями. Основной функцией БСС, наряду со сбором информации об окружающей среде, является ее передача. Механизмы обмена сообщениями критически важны для БСС.
- Режимы функционирования. Из-за необходимости экономии энергии моты должны функционировать в различных режимах, предусматривающих различные схемы питания их компонентов. Таких режимов у одного устройства может быть несколько.
- Работа на основе событий. В работе БСС важную роль играют события, то есть изменения, происходящие в окружающей среде или в самом устройстве, на которые должна следовать определенная реакция. События обычно связаны с работой устройств мота, например событие «получено сообщение» инициируется приемопередатчиком, а событие «получены показания» — датчиком. Следует выделить события, возникающие при ошибках в функционировании компонентов мота.
- Псевдопараллелизм. В связи с ориентацией на обработку событий моты работают по принципу псевдопараллелизма: в составе управляющей программы выделяются задачи, которые выполняются единственным процессором последовательно, однако их выполнение может прерываться обработкой событий. Таким образом, мот постоянно готов к обработке событий, каждое из которых предполагает свою последовательность действий. Эти действия, разделенные на задачи, будут выполняться попеременно в соответствии со своим приоритетом, что на определенном уровне абстракции можно считать одновременным их исполнением.
- Различные уровни абстракции. Различные платформы предоставляют конечному пользователю различные возможности для настройки сети на конкретное приложение.

Рассмотрев основные характеристики предметной области, учет которых требуется при разработке управляющих приложений для БСС, перейдем к обсуждению возможностей, предоставляемых платформой моделирования SQL Server Modeling.

Моделирование и платформа SQL Server Modeling

Вначале заметим, что моделирование уже давно играет важную роль в разработке ПО, позволяя представлять структуру и поведение будущей программы в виде удобной для обсуждения, изменения и воплощения схемы-модели. Модели используются для разработки вариантов воплощения той или иной необходимой функциональности, описания процессов, потоков данных и т. д., а также для закрепления принятых решений, общения специалистов из разных областей и представления структуры будущего или уже созданного ПО в формальном виде. Таким образом, модели используются на всех этапах разработки ПО, создаются же они, в основном, на этапе проектирования.

Наиболее известным языком моделирования является UML (Unified Modeling Language). Он широко применяется для описания ПО в различных областях, однако из-за своей универсальности имеет ряд недостатков. Создание собственных средств моделирования, предназначенных для работы в конкретной предметной области, значительно повышает скорость и качество разработки соответствующих программных продуктов.

Процесс разработки модели, ориентированной на конкретную предметную область, называется предметно-ориентированным моделированием (Domen Specific Modeling), а язык, которым оперирует разработанная модель, — предметно-ориентированным языком (Domen Specific Language, DSL). Предметно-ориентированные языки могут быть текстовыми и графическими. Целесообразно использовать DSL в случаях, когда разработчикам приходится работать в определенной предметной области или с применением технологий, для описания которых языков моделирования общего назначения недостаточно. Именно такая ситуация сложилась в области БСС.

Метод создания приложений, при котором в качестве спецификаций используются модели сущностей, процессов и данных предметной области, получил специальное название — разработка, основанная на моделях. Этот метод предусматривает:

- использование абстракций для представления структуры на необходимом уровне детализации, что подразумевает исключение из модели деталей, учет которых на данном этапе не является необходимым;
- использование моделей и отношений как основы разработки;
- использование модельно-ориентированных компонентов, которые соответствуют требованиям приложения или бизнес-процесса;
- связывание моделей и моделируемых сущностей на всех стадиях разработки таким образом, чтобы в течение всего жизненного цикла эти связи сохранялись;
- автоматизацию средств разработки и создание артефактов таким образом, чтобы обеспечивалась возможность их повторного использования.

Основным достоинством такого подхода является поддержка эффективного и легко управляемого процесса создания сложных приложений.

Платформа SQL Server Modeling (прежнее название OSLO), в настоящее время разрабатываемая компанией Microsoft, специально приспособлена для описанного выше подхода. Целью создания платформы SQL Server Modeling является преодоление разрыва между идеями разработчиков и реальными программными компонентами, которые для реализации этих идей разрабатываются и используются в сложных распределенных приложениях. Моделирование приложений означает представление описаний и спецификаций в виде унифицированных данных и схем, которые значительно удобнее в восприятии, чем нестандартизированные описания, то есть являются удобными средствами представления результатов работы и надежными средствами коммуникации между разработчиками и заказчиками.

Предметно-ориентированный язык

С учетом всех рассмотренных выше тенденций и особенностей предметной области авторами статьи был разработан язык описания функционирования БСС. Предлагаемый язык предназначен для описания функционирования узлов беспроводной сенсорной сети. Если сеть является гетерогенной, т. е. содержит узлы различной архитектуры, то для ее описания необходимо составить схемы работы узлов всех входящих в нее типов.

Предлагаемый язык не предназначен для описания работы базовой станции и построения топологии сети. Перечисленные возможности не нужны для описания функционирования БСС, так как базовая станция, являясь интерфейсом между беспроводной сенсорной сетью и потребителем информации, чаще всего представлена стандартным компьютером, снабженным необходимым приемопередатчиком, что позволяет описывать его функционирование обычными средствами. Архитектура сети предполагается выбранной при построении схем функционирования БСС и для предлагаемого языка является внешней информацией, используемой в нем опосредованно. Так, при описании однородной сети схемы функционирования всех узлов совпадают. Если же в сети предусмотрены узлы-маршрутизаторы, то для них потребуются особые схемы. Аналогично, использование иерархической структуры с возможностью передачи функций маршрутизатора найдет отражение в описании состояний узлов. Способности мотов к самоорганизации обеспечивают создание сети любой допустимой конфигурации без изменения схем функционирования узлов. Однако необходимость учета расположения узлов может привести к созданию отдельных схем функционирования для разных узлов.

Предложенный язык имеет три уровня описания, которые представлены схемами различной нотации. Первый из них — компонентный. Он является базовым для создания остальных схем. Компонентный уровень описывает устройства, из которых состоит мот, и их особенности, такие как возможные состояния, принимаемые команды, возвращаемые события, ошибки и т. д. Также на данном уровне описываются необходимые структуры памяти (предназначенные для хранения списков соседей, сообщений, данных) и типы сообщений, отправляемых и принимаемых узлом.

Второй уровень — уровень состояний. На данном уровне описываются состояния мота, зависящие от состояний его компонентов и выполняемых алгоритмов. Считается, что в каждом состоянии выполняется единственный алгоритм и компоненты узла остаются в фиксированном состоянии. Последнее условия не является обязательным, однако удобно в работе. Переходы между состояниями осуществляются после завершения или при прерывании соответствующих алгоритмов: это может быть выход по событию, по выполнению необходимых действий, по ошибке и т. д.

Третий уровень — уровень функционирования. Он описывает работу мота, находящегося в конкретном состоянии. Описание функционирования состоит из ветвящейся, в зависимости от условий, нити управления, с которой связаны действия и события. Действия — это запрограммированные операции, выполняемые мотом. События определяются в соответствии с терминологией, принятой в БСС. Кроме того, данный уровень учитывает обращения к компонентам узла и работу с сообщениями, описанными на компонентном уровне. Для лучшей читаемости схем данного уровня, которые могут быть достаточно сложными, допустима вложенность схем друг в друга. То есть действие может описываться дополнительной схемой. Такие действия названы действиями, требующими детализации. Не требуют детализации действия, суть которых очевидна на данном уровне абстракции, либо действия, являющиеся командами языка управления мотами, например NesC.

Полный набор схем, описывающий некоторое приложение, называется функциональной схемой этого приложения.

Для удобства работы с предлагаемым языком он представлен в двух нотациях — графической и текстовой. Рассмотрим его основные возможности на примере проектирования реальной программы, создаваемой на языке программирования БСС NesC. Программа управляет сбором и передачей данных в однородной сети, то есть в такой сети, где все узлы одинаковы и выполняют одинаковые алгоритмы. Программа использует протоколы передачи данных из библиотеки XMesh, описание которых не приводится.

Первым уровнем описания является компонентный.

На схеме (рис. 1) представлены все компоненты мота — таймер, батарея, сенсор, набор индикаторов, приемопередатчик, память — с соответствующими им командами, событиями, состояниями и ошибками. Также на этой схеме присутствуют необходимые для работы структуры памяти — список показаний, флаг памяти, флаг критичности, флаг передачи и их характеристики (поля, размеры, команды). К типам принимаемых и передаваемых сообщений относится единственное присутствующее на схеме сообщение.

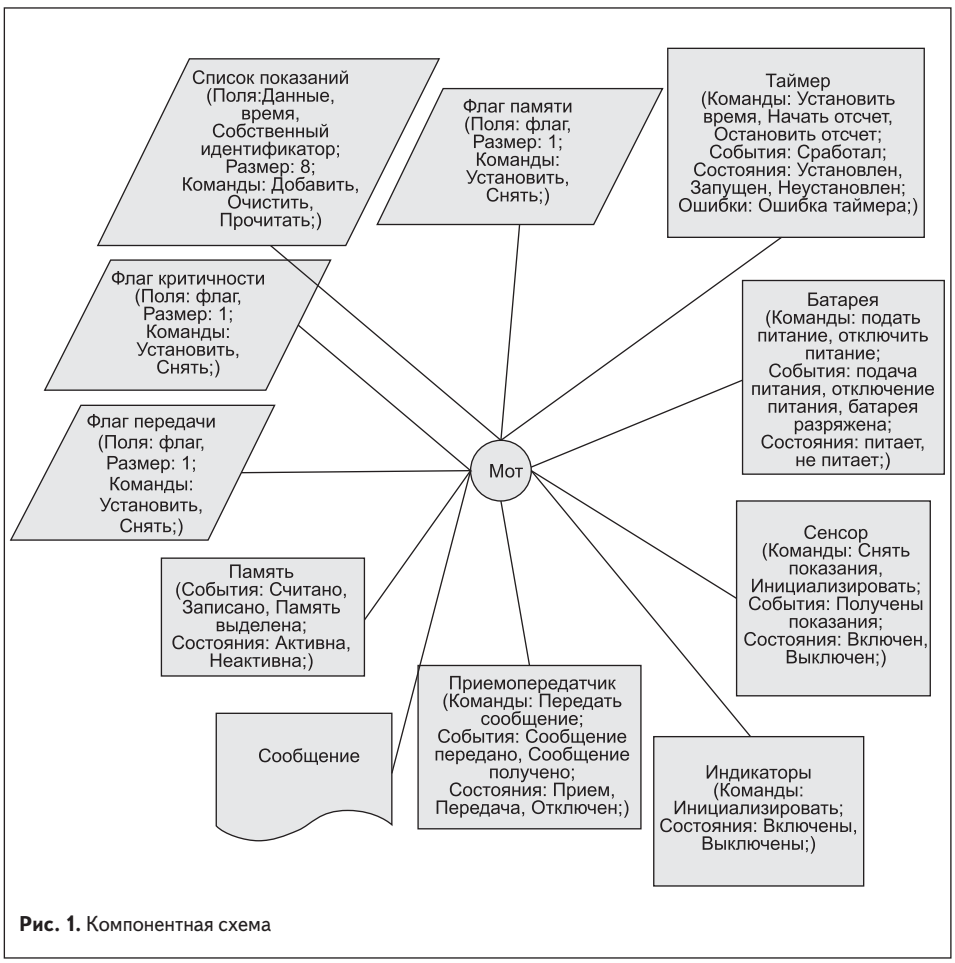

Рис. 1. Компонентная схема

В текстовой нотации представленная схема выглядит достаточно объемно, поэтому приведем только ее часть (врезка 1).

Врезка 1

Описание компонента «Таймер»
Данный компонент выполняет команды:
«Установить время», «Начать отсчет»,
«Остановить отсчет».
Данным компонентом инициируется событие
«Сработал».
Данный компонент может находиться в одном из состояний:
«Установлен», «Запущен» или «Не установлен».
Для данного компонента возможны ошибки:
«Ошибка таймера».
...
Описание структуры памяти «Список показаний»
Данная структура имеет поля:
«Данные», «Время» и «Собственный идентификатор».
Размер данной структуры: 8 **элементов.**
Для данной структуры применимы команды:
«Добавить», «Очистить», «Прочитать».
Описание структуры памяти «Флаг передачи»
Данная структура имеет поле «Флаг».
Размер данной структуры: 1 **элемент.**
Для данной структуры применимы команды:
«Установить», «Снять».

Далее следует описание уровня состояний (рис. 2).
Управляемый описываемой программой узел может находиться в одном из трех состояний. «Отключен» — питание отключено, мот не активен. «Инициализация и запуск программы» — после подачи питания мот переводится в рабочее состояние. «Работа» — непосредственно функционирование мота. Переходы между состояниями происходят по событиям подачи или отключения питания и по окончании выполнения алгоритма инициализации.

Рис. 2 Схема состояний

В текстовой нотации часть данной схемы выглядит следующим образом (врезка 2):

Врезка 2

Описание схемы «Тестовая программа»
Работа начинается с состояния «Отключен».
...
Описание состояния «Инициализация и запуск программы».
Компоненты находятся в состояниях:
«Память» — «Активна»,
«Сенсор» — «Отключен», «Приемник» — «Прием»,
«Батарея» — «Подключена».
Из данного состояния возможны переходы:
в состояние «Работа» **по результату**
«Инициализация окончена»,
в состояние «Отключен» **по результату**
«Питание отключено».
...

Уровень функционирования содержит не менее одной схемы для каждого состояния, присутствующего в схеме уровня состояний. Однако их может быть и больше, если в основных схемах встречаются действия, требующие детализации.
В рассматриваемом нами примере обратим внимание на одну из схем функционирования — описание алгоритма «Работа».
На рис. 3 кругами с тенью обозначается начало и конец алгоритма; обычными кругами — события компонентов мота, описанные в компонентной схеме; скругленными прямоугольниками — действия; скругленными прямоугольниками с тенью — действия, требующие пояснений; параллелограммами — структуры памяти, участвующие в действии. Стрелки обозначают нити управления, треугольники — выбор «исключающее или». Вертикальная черта соответствует псевдопараллельной обработке следующих за ней событий. На рисунке не представлена работа с сообщениями: она описывается так же, как и работа со структурами памяти (сущность «сообщение» обозначается, как на компонентной схеме).
Приведем часть описания в текстовой нотации (врезка 3):

Врезка 3

Описание алгоритма «Работа»
Вначале происходит параллельное выполнение
«Ожидание событий».
Параллельное исполнение «Ожидание событий»
позволяет обрабатывать события «Сработал»
(таймер показал, что наступило время снимать показания),
«Память выделена», «Показания получены»,
«Записано»,
«Отключение питания».
Событие «Сработал» **инициируется компонентом** «Таймер».
После данного события делается проверка
«Выделена ли память».
Проверка «Выделена ли память» **предполагает следующие варианты:**
либо «Память выделена»,
тогда выполняется действие «Снятие показаний»,
либо «Память не выделена»,
тогда происходит параллельное выполнение
«Ожидание событий».
...
Данный алгоритм может завершиться переходом в состояние «Отключен».

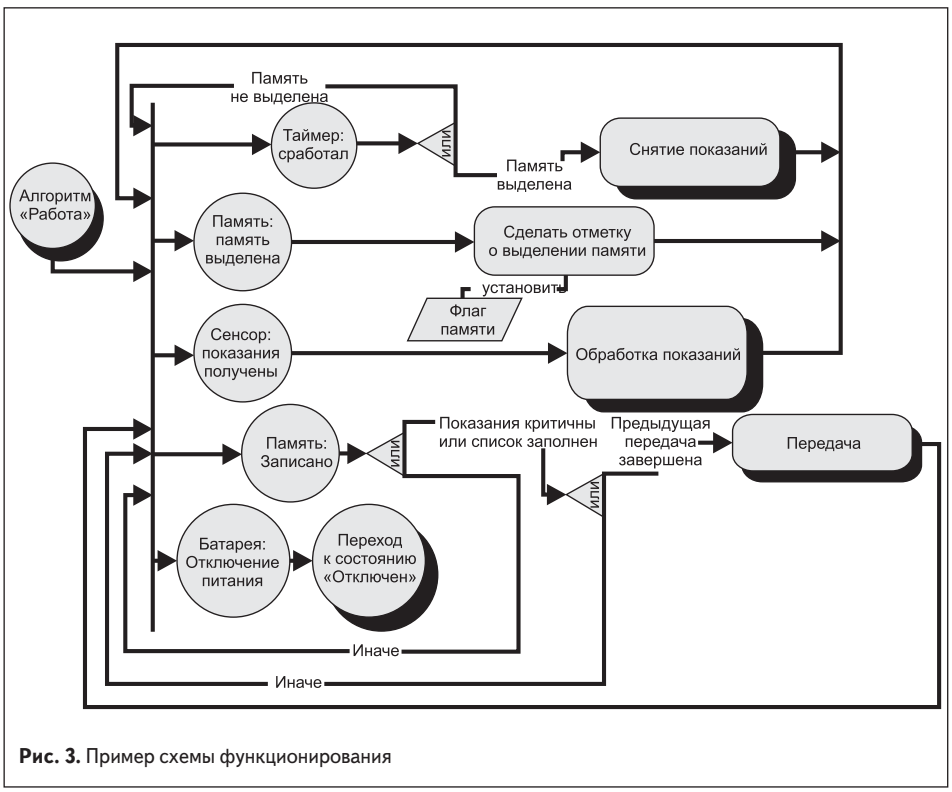

Рис. 3. Пример схемы функционирования

Заключение

Мы описали предметно-ориентированный язык, направленный на создание моделей приложений для БСС и учитывающий принятые в БСС стандарты. Разработаны две нотации языка — графическая и текстовая. Проанализированы возможности применения платформы Microsoft SQL Server Modeling для создания средства моделирования приложений для БСС.

В процессе разработки предметно-ориентированного языка были использованы сущности, участвующие в функционировании приложения на узле БСС, а также их характеристики: состояния, команды, сообщения и т. д. Для наиболее полного и удобного описания приложений в спецификации языка выделены три следующих уровня:

- Компонентный уровень описывает физические и программные компоненты устройства, а также необходимые структуры памяти и шаблоны используемых сообщений.
- Уровень состояний рассматривает состояния мота в целом, часто соответствующие режимам его питания; состояние мота характеризуется состояниями его компонентов и выполняемым в данном состоянии алгоритмом.
- Уровень функционирования описывает конкретные последовательности операций, выполняемых мотом. Схемы уровня функционирования могут расширять и дополнять друг друга для описания приложений любой сложности в доступной восприятию форме.

Использование трех уровней языка дает возможность сформировать полное описание функционирования мота, позволяя разрабатывать соответствующие алгоритмы любой степени абстракции — от непосредственного управления аппаратной частью до работы с высокоуровневыми программными компонентами. Работа с упомянутыми высокоуровневыми компонентами происходит через описанные для них интерфейсы по схеме «команда–ответ», принятой в БСС. При этом реализация используемых программных компонент для предлагаемого языка не важна.

В настоящее время ведется разработка программного инструментального средства на платформе SQL Server Modeling, которое позволит строить схемы на предметно-ориентированном языке и транслировать их в фрагменты управляющего кода для исполнения непосредственно на мотах.

Поисковая научно-исследовательская работа выполнена в рамках федеральной целевой программы «Научные и научно-педагогические кадры инновационной России».