УДК 551.46.077:629.584

ВИЗУАЛЬНАЯ СРЕДА РАЗРАБОТКИ ЗАДАНИЙ ДЛЯ АВТОНОМНЫХ ПОДВОДНЫХ РОБОТОВ

А.В. Инзарцев, В.Ю. Матвиенко

Институт проблем морских технологий ДВО РАН, Владивосток

Описывается система планирования миссий для автономных необитаемых подводных аппаратов. Приводится краткий обзор современных средств визуального программирования, применяемых для составления заданий для роботов. Главным образом внимание уделяется средствам, основанным на парадигме программирования потоков данных. Перечислены функциональные возможности, технологии планирования и представления задания, основные архитектурные принципы создаваемой системы.

ВВЕДЕНИЕ

Область применения современных автономных необитаемых подводных аппаратов охватывает разнообразные задачи изучения морских глубин, поиска и обследования объектов на дне, картографической съемки, обеспечения безопасности акваторий и многие другие. Программное обеспечение такого многоцелевого подводного робота включает широкий набор средств, функционирующих как в бортовой вычислительной сети аппарата, так и на стороне оператора. Программа-задание (миссия), выполняемая в системе управления, является по существу компонентом, который решает текущую прикладную задачу, используя все доступные возможности аппарата. Планирование миссии оператором непосредственно перед запуском в условиях ограниченного времени требует средств автоматизации, обладающих следующими свойствами:

- способностью адаптироваться к различным конфигурациям АНПА;
- простотой в использовании человеком, не имеющим программистской подготовки;
- достаточной гибкостью для решения множества задач;
- возможностью моделировать выполнение задания.

МЕТОДЫ ПЛАНИРОВАНИЯ ЗАДАНИЙ ДЛЯ РОБОТОВ

Применяемые способы описания заданий

В том или ином виде программа-задание присутствует в составе программного обеспечения большинства автономных мобильных систем. В настоящее время приняты следующие средства ее представления и размещения в системе управления.

Исполняемый модуль. Наиболее распространенный вариант, при котором код задания разрабатывается в той же программной среде, что и система управления (как правило, на одном из широко применяемых языков программирования), и представляет собой независимую программу, использующую функциональность системы управления. В частности, в разработках ИПМТ ДВО РАН программа-задание использует так называемую библиотеку команд АНПА, предоставляющую набор процедур для высокоуровневого управления поведением робота.

Интерпретируемый скрипт. Последнее время в программной инженерии широкую популярность приобретают подходы, связанные с разработкой узко специализированных языков для решения различных прикладных задач (так называемые Domain Specific Languages [1,2]). Среди языков программирования роботов можно отметить такие, как URBI [3], Microsoft Visual Programming Language (см. ниже). Реализация этого метода требует наличия на борту аппарата интерпретатора языка высокого уровня абстракции, что может быть проблематично для низкопроизводительных систем.

Транслируемый скрипт. Данный подход, совмещая два предыдущих, предполагает использование на первом этапе формирования абстрактного описания конфигурации и поведения робота (например, с помощью многоагентной системы конечных автоматов [4]) и последующую его трансляцию в исполняемый код на языке общего назначения.

Средства и методы автоматизированного планирования

Средства автоматизации планирования задания находят широкое применение в первую очередь в силу того, что сокращают трудоемкость этого

процесса. Можно выделить 2 типа подобных систем:

- основанные на универсальных средствах визуального программирования;
- специализированные.

Универсальные средства визуального программирования

В первом случае методика составления задания предполагает использование визуальных языков общего назначения для разработки управляющей программы. Подавляющее большинство подобных систем основано на парадигме программирования потоков данных (Data Flow Programming [5]). Ключевая идея этого метода заключается в явном представлении потоков данных, обеспечивающих обмен сообщениями между элементами системы. Элементы (операции или узлы в графических средах) характеризуются набором входящих и исходящих потоков данных и реакцией на получаемые сообщения, которая может выражаться в выполнении сложной процедуры, описанной на языке программирования общего назначения. Таким образом, программа на визуальном языке представляет собой схему потоков данных между операциями предметной области. В качестве примера можно привести краткое описание двух наиболее известных свободно распространяемых систем такого рода.

■ ROBOTFLOW

Проект RobotFlow [7] (рис.1) представляет собой расширение среды визуального программирования FlowDesigner и входит в состав крупнейшего свободно распространяемого пакета программного обеспечения для роботов MARIE, объединяющего около 20 проектов с открытым исходным кодом.

Среда позволяет составлять программы на C++ с помощью манипуляций над диаграммами потоков данных. Типичный графический примитив (узел) представляет собой функцию, имеющую входные и выходные параметры (соответственно входы и выходы узла). Объединение узлов в сети позволяет решать широкий класс задач, связанных со сложной обработкой потоков данных, такой как обработка сигналов или анализ изображений. Предусмотрены два типа узлов: базовые и полученные в результате композиции нескольких узлов в сеть.

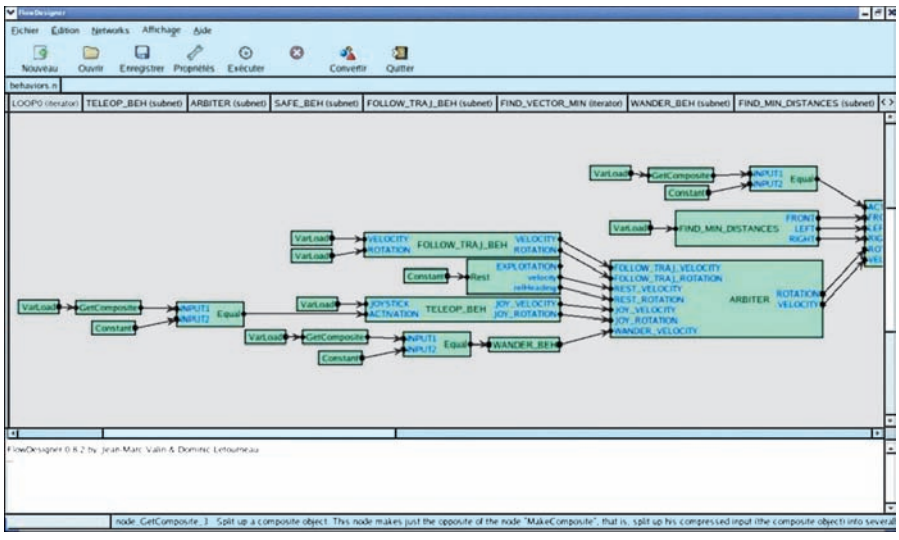

Рис. 1. Система планирования заданий RobotFlow

Архитектура программы позволяет разрабатывать дополнительные модули для расширения множества базовых узлов.

Среда обладает возможностями визуальной отладки программы, представляемой полученной сетью узлов. Поддерживаются специализированные редакторы и визуализаторы для различных типов данных. Например, если на каком-либо этапе на вход одного из узлов поступает изображение, программист в процессе отладки может его просмотреть или изменить в специализированном графическом редакторе. Допускается развитие этой технологии для других типов данных, например для отображения положения робота на карте.

■ MICROSOFT ROBOTICS STUDIO

Бесплатная для некоммерческого использования среда разработки программного обеспечения роботов от Microsoft предлагает архитектуру, основанную на сервисах, схожую в отношении подхода к планированию и симуляции миссии с RobotFlow. Под сервисом понимается независимый компонент системы, решающий какую-либо задачу обработки данных или управления. Возможно изменение конфигурации таких компонентов в процессе работы системы. В качестве средства описания задания используется Microsoft visual programming language [6] (рис. 2), основанный на xml и представляемый графически в виде вышеупомянутых диаграмм потоков данных.

Microsoft robotics studio кроме инструментов разработки и исполняющей среды включает среду симуляции и отладки, поддерживающую современные средства 3D визуализации и моделирования, в том числе аппаратные ускорители физических расчетов.

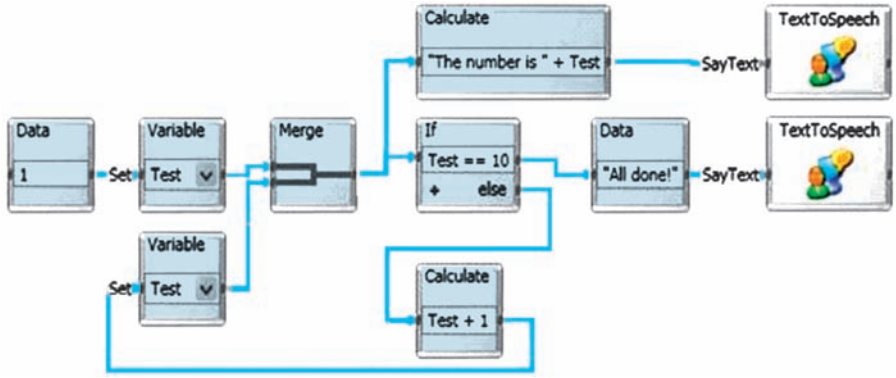

Рис. 2. Microsoft Visual Programming Language

■ ОБЩИЕ ЧЕРТЫ РАССМОТРЕННЫХ СИСТЕМ

Главным достоинством рассматриваемых систем можно считать их универсальность. Парадигма программирования, ориентированная на потоки данных, находит применение далеко за рамками программирования роботов. В научно-исследовательской среде широко известны такие программные пакеты, как LabView, MatLab Simulink, позволяющие визуальными средствами решать полный спектр задач моделирования. Универсальность рассматриваемых систем обеспечивается в первую очередь за счет унифицированного графического представления. Нотация диаграмм потоков данных позволяет визуально редактировать и отлаживать любую программу, представленную в терминах операций над потоками данных. Отладка в данном случае подразумевает отслеживание сообщений, передаваемых между элементами в процессе выполнения программы. Немаловажно также, что рассматриваемые среды программирования допускают расширение набора используемых операций за счет пользовательских функций, что является свидетельством их применимости практически для неограниченного множества задач.

Специализированные системы

Универсальным характером вышеописанных визуальных сред обусловлены не только их положительные стороны, но и существенные недостатки связанного с ними подхода. Как и любой многоцелевой инструмент, они не всегда являются наиболее эффективным способом решения каждой отдельно взятой задачи. Скажем, для оператора АНПА более удобным является способ составления задания, позволяющий прокладывать маршрут движения на карте, управляя поведением отдельных устройств и систем, по сравнению с редактированием программы, представленной на диаграмме.

Иными словами, существует необходимость в системах планирования миссий, ориентированных на конкретную программную архитектуру и функциональные возможности робота. Следует заметить, что подробное рассмотрение систем такого рода не представляет особого интереса, поскольку их внедрение на практике неосуществимо. Главным образом это связано с тем, что они являются неотъемлемой частью соответствующих программно-аппаратных комплексов и не распространяются отдельно. Другой причиной является необходимость приложения неоправданных усилий для включения специализированных систем в комплекс программного обеспечения, на которое они изначально не были рассчитаны. Кроме того немногочисленные научные статьи и материалы конференций на эту тему (например, [8], [9]) носят в основном обзорный характер, не раскрывая деталей реализации описываемых систем.

■ ИНСТРУМЕНТАРИЙ ПОДГОТОВКИ ЗАДАНИЙ ДЛЯ АНПА ИПМТ ДВО РАН

В ИПМТ ДВО РАН исторически сложился подход к планированию миссий АНПА с использованием визуального редактора собственной разработки [10], обладающего преимуществами, свойственными специализированным средам. В частности, система ориентирована не на редактирование программы как таковой, а на манипулирование предполагаемым результатом ее выполнения (траекторией АНПА на карте). Для установки режима работы бортовых устройств использовался специализированный редактор. Код управляющей программы представлял собой последовательность известных редактору команд.

Несмотря на высокую эффективность системы на начальном этапе, в процессе более десяти лет использования был выявлен ряд недостатков, затруднявших ее дальнейшее применение и сделавших нецелесообразным расширение. Сказался монолитный характер архитектуры системы, не позволяющий без существенной переработки добавлять такие функции, как поддержка составных команд и возможность планирования многовариантных миссий. Кроме того, сильно устарели и вышли из употребления примененные технологии и средства разработки.

В настоящее время в ИПМТ ДВО РАН с учетом современных тенденций развития средств визуального программирования, а также опыта использования и проектирования предшествующей системы планирования миссий разрабатывается среда МПЛАН, общая структура которой изображена на рис.3. Далее пойдет речь о некоторых возможностях системы и принятых подходах.

Описание задания

На начальном этапе разработки системы планирования важно определить, в какой форме будет представлено задание. При использовавшемся прежде подходе заданием считался исходный код управляющей программы на языке С, составленный с соблюдением определенных правил и ограничений на использование языковых средств, а также снабженный комментариями, несущими информацию для редактора. В среде МПЛАН принято разделение представления задания на исполняемое и удобочитаемое при поддержке совместимости с унаследованным форматом.

Для представления задания в виде, легком для восприятия человеком (в т.ч. нетехническим пользователем), разработан специализированный расширяемый язык на базе xml, состоящий как из команд управления АНПА (используемые команды можно найти, например, в [10]), так и из директив управления средой редактирования и моделирования миссии. Это позволило не только упростить описание составляемой миссии, но и обеспечивать проверку синтаксической корректности не на уровне конструкций языка С, как делалось прежде, а на существенно более высоком уровне абстракции.

В результате работы редактора генерируется код управляющей программы - форма задания, предназначенная для непосредственного исполнения аппаратом.

Редактирование задания

Наличие полноценного текстового описания полезно для быстрого создания элементарных миссий в обычном текстовом редакторе, однако основной задачей при разработке системы МПЛАН было предоставление пользователю развитых средств визуального редактирования. Осознавая важность сохранения простоты интерфейса исходной системы, требовалось обеспечить следующие принципиально важные функциональные возможности:

- оперативное добавление пользователем в систему новых команд;
- планирование различных вариантов миссий в зависимости от условий времени выполнения.

Существенная сложность в решении этих задач заключается в необходимости предусмотреть возможность как моделирования выполнения команд, не известных на этапе разработки системы, так и редактирования результатов их выполнения. Кроме того большое значение имеет способ описания команд, от которого требуется высокая степень выразительности, чтобы не вводить лишних ограничений на множество допустимых заданий.

В среде МПЛАН предусмотрено две категории команд: базовые и составные. В первую попадают команды, непосредственно реализуемые системой управления аппарата и моделируемые средой, а также оператор условного перехода. Вторую составляют сложные процедуры на языке С, составленные из базовых команд и добавляемые пользователем в процессе планирования задания. Этой второй группой определяется существенная гибкость способа описания команд.

Для того чтобы обеспечить возможность моделирования результатов выполнения задания и их корректировки, вычислительный процесс, порождаемый исполнением последовательности команд, дискретно представляется в виде дерева переходов АНПА из одного состояния в другое с ветвлениями в зависимости от условий рабочего цикла программы.

Под состоянием АНПА понимается вектор параметров, включающих характеристики движения аппарата, статус бортовых устройств, некоторые существенные переменные системы управления и т.д. Важно отметить, что данная структура может быть легко расширена при необходимости более де-

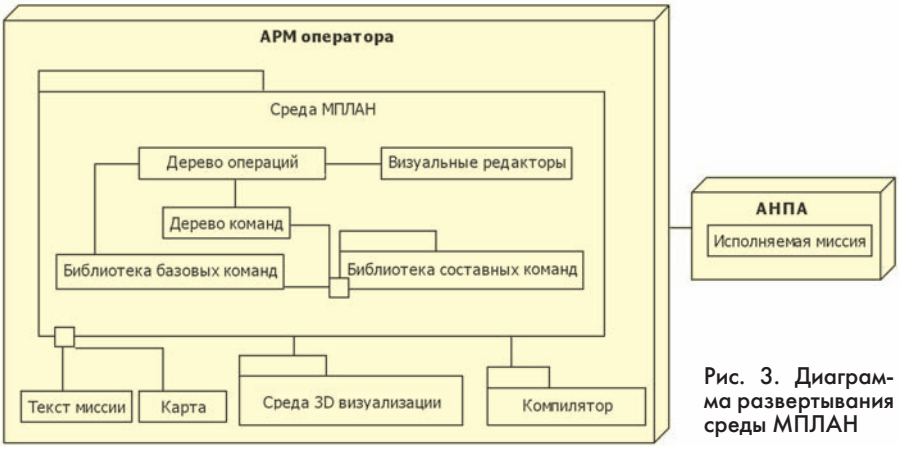

Рис. 3. Диаграмма развертывания среды МПЛАН

тального моделирования поведения АНПА.

Переход между состояниями в результате выполнения базовой команды называется операцией. Иными словами, операция — это шаг выполнения программы. В терминологии потоков данных операция получает на вход состояние АНПА и аргументы соответствующей команды и выдает на выходе новое состояние АНПА (рис.4).

Изменение каждой операции возможно при помощи универсального диалогового редактора параметров связанной с ней команды. Кроме того разработаны специализированные средства редактирования и визуализации для различных аспектов выполнения задания. Например, изменение пути движения аппарата возможно посредством прокладки траектории на карте, а вертикальный профиль движения аппарата отображается на отдельной диаграмме.

Здесь важно отметить принципиальное отличие в решении задач моделирования выполнения команды и редактирования.

В первом случае для каждой команды достаточно определить преобразования модели аппарата. В среде МПЛАН это достигается по-разному для разных видов команд. Базовые команды добавляются в систему сопровождающим программистом, т.е. их поведение жестко закодировано в среде редактора. Сложные команды (например, движение аппарата по спирали) разрабатываются пользователем системы и выполняются редактором непосредственно с учетом базовых команд, из которых они состоят, что позволяет автоматически обеспечивать моделирование любой новой команды.

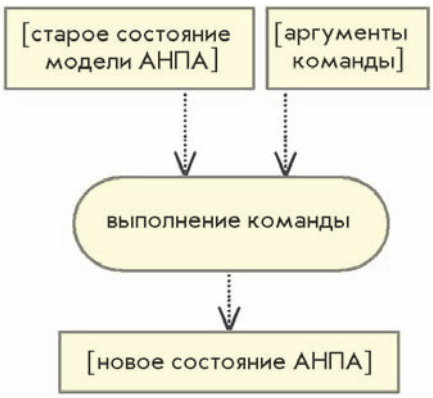

Рис. 4. Структура операции

Редактирование возможно явным заданием параметров команды, однако такой метод не всегда соответствует критериям простоты, поэтому для большинства используемых команд разработаны обратные преобразования, позволяющие, изменяя предполагаемый результат выполнения (например, положение целевой точки на карте), косвенно подбирать необходимые значения аргументов.

Функциональность и возможности расширения

При разработке оконного интерфейса существенное внимание уделялось тому, чтобы максимально упростить расширение представлений модели выполняемой миссии. В настоящее время допускаются несколько эквивалентных форм планирования задания. Пользователь может решить одну и ту же задачу:

- при помощи встроенного текстового редактора;
- используя визуальный редактор дерева команд;
- оперируя элементами траектории на географическом планшете.

Последний способ является наиболее предпочтительным для большинства пользователей. Оператор может редактировать множество возможных траекторий и атрибутов состояния аппарата на планшете (рис. 5), тем самым изменяя параметры соответствующих команд. К примеру, можно выбрать на планшете участок траектории (галс) и переместить его целевую точку, в результате чего будет найдена команда, соответствующая данному участку, рассчитаны и установлены соответствующие параметры движения. Внесенные изменения в план движения аппарата отображаются в дереве команд и текстовом представлении миссии. Для отображения положения АНПА по высоте и глубине на всей траектории движения используется окно вертикального профиля, представляющее собой диаграмму глубин и высот движения аппарата. Окно конфигурации позволяет редактировать параметры географической привязки карты.

Необходимость предусмотреть многочисленные возможности расширения системы диктуется изменчивостью конфигураций АНПА и решаемых задач. Проектирование системы в строгом соответствии с принципами объектно-ориентированного программирования позволяет в значительной степени независимо изменять и дополнять модель АНПА, классы команд и визуальных редакторов. К примеру, для добавления базовых команд с минимальными усилиями со стороны программиста создан отдельный редактор метаданных, генерирующий большую часть кода соответствующих классов по описанию команды.

Возможности системы также расширяются за счет новых средств, облегчающих планирование задания в реальных условиях. Возможно отображение реальной траектории движения аппарата, получаемой из бортового накопителя данных. Одной из важных функций является также сохранение дерева операций и его передача в систему трехмерной визуализации, проигрывающей движение аппарата над донным ландшафтом [12].

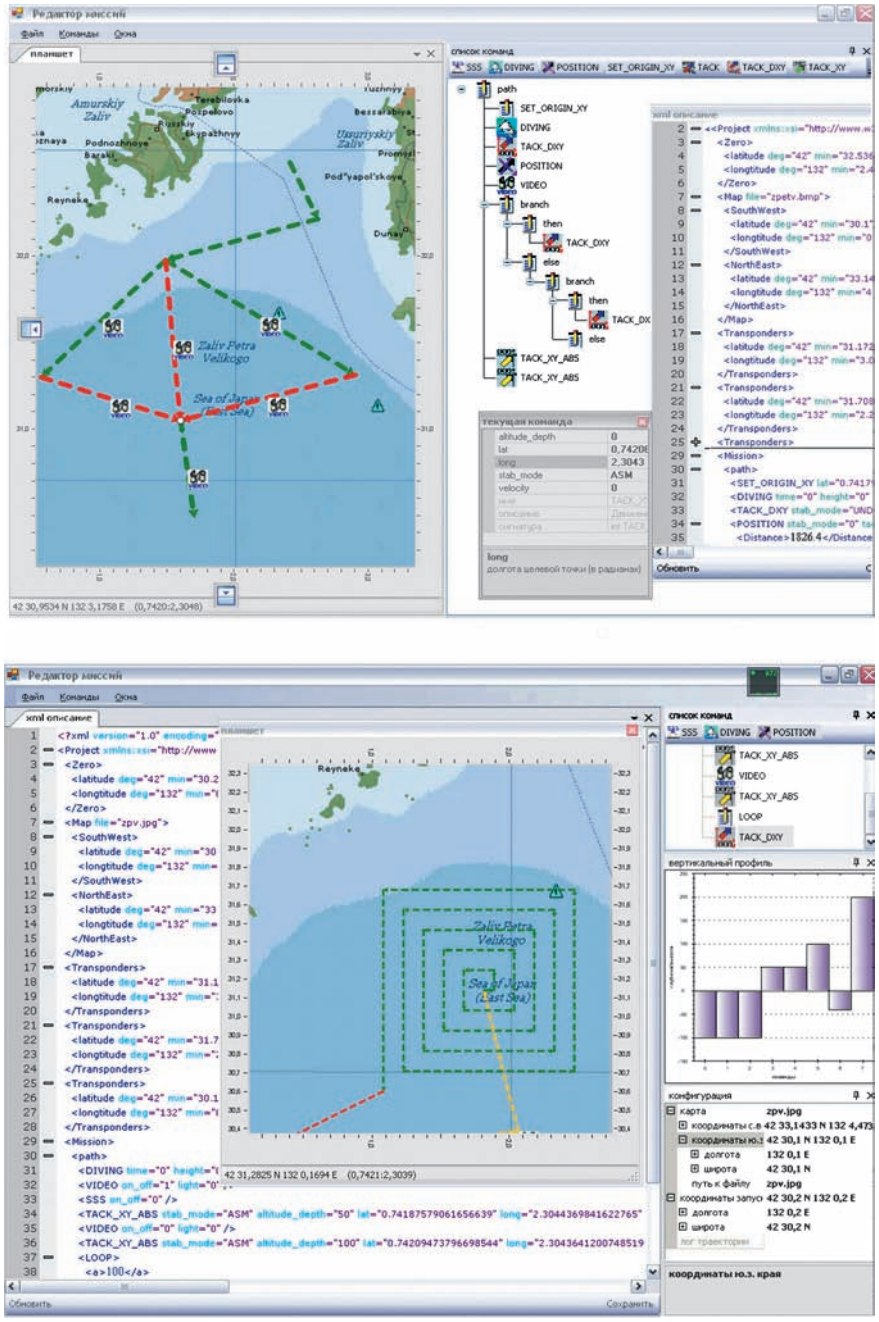

Рис. 5. Интерфейс среды планирования миссий МПЛАН

ЗАКЛЮЧЕНИЕ

Рассмотренные возможности системы позволяют существенно упростить процесс планирования заданий для АНПА и как следствие повысить его эффективность. Отметим некоторые нерешенные задачи и перспективы развития системы. Среди неявных допущений, принятых при разработке системы, было предположение корректного завершения команд, что на практике не всегда оправдано. В связи с этим важным направлением развития системы является обеспечение возможности самостоятельного перепланирования задания аппаратом в процессе работы в случае возникновения непредвиденных ситуаций. Определенные шаги в этом направлении предпринимались и ранее [10].

Решение этой задачи связано прежде всего с необходимостью планирования реакции на асинхронные события, что требует дальнейшей, более детальной разработки предпринятого подхода к моделированию задания.

Существенным шагом в развитии системы представляется ее включение в состав разрабатываемого моделирующего комплекса, имитирующего полный цикл управления аппаратом от получения сенсорных данных и моделирования движения до выполнения введенного задания [11].

ЛИТЕРАТУРА

1. Fowler. M. Domain Specific Languages. http://martinfowler.com/bliki/DomainSpecificLanguage.html.
2. Raymond E.S. The Art of Unix Programming. Addison-Wesley, 2003.
3. Baillie J.C. URBI Language Specification, 2005.
4. MissionLab User Manual, Georgia Institute of Technology, Atlanta. http://www.cc.gatech.edu/ai/robot-lab/research/MissionLab/.
5. Sutherland W.R. The On-line Graphical Specification of Computer Procedures // MIT. 1966.
6. Microsoft Robotics Studio Visual Programming Language. http://msdn.microsoft.com/robotics/learn/default.aspx#vpl.
7. Robot Flow, Mobile Robotics Toolkit for FlowDesigner Manual. http://robotflow.sourceforge.net/.
8. Woodrow I. Autonomous AUV mission planning and replanning. Systems Engineering & Assessment ltd. 10p.
9. Christiane N., Duarte A. Common control language for dynamic tasking of multiple autonomous vehicles. Naval Undersea Warfare Center. 8p.
10. Автономные подводные роботы: системы и технологии / М.Д. Агеев, Л.В. Киселев, Ю.В. Матвиенко и др.; Под общ. ред. акад. М.Д.Агеева. М.: Наука, 2005. 398 с.
11. Инзарцев А.В., Киселев Л.В.,.Бобков В.А и др. Имитационный моделирующий комплекс для «интеллектуального» автономного подводного робота // Мехатроника, автоматизация, управление. 2008. №7 (в печати).
12. Багницкий А.В. Программная среда для трёхмерного представления миссии подводного аппарата // Матер. VI Всерос. науч.-практ. конф. студентов, аспирантов и молодых ученых. Томск, 2008.