Специфика разработки программного обеспечения для систем управления технологическим оборудованием в реальном времени

Мартинов Г.М., Мартинова Л.И., Григорьев А.С.,
МГТУ "Станкин"

Ключевые технологии разработки. Невозможно приступать к реализации сложного проекта, рассчитанного на несколько человеко-лет, без проектной документации. Сегодня стандартом де-факто стал язык UML, который используют для создания программных моделей. В такие модели легко вносить функциональные изменения, связанные с эволюцией системы управления (рис. 2).

Повторное использование программного кода, независимого от конкретной аппаратной реализации, осуществляют с помощью компонентного COM-подхода, используемого согласно строго определенной схеме.

Используя готовые механизмы, предоставляемые MS XML и Internet-технологиями, можно расширить функциональные возможности системы управления, повысить ее надежность и при этом сократить время разработки системы.

Компонентная архитектура. Схему построения компонентной архитектуры рассмотрим на примере приложений логического анализатора (рис. 2), который отображает входы и выходы системы управления электроавтоматикой и используется для отладки PLC-программ. Здесь компонентный подход предполагает выделение двух уровней абстракции. Абстракция на уровне COM-сервера маскирует аппаратные особенности оборудования. Этот уровень обеспечивает совместимость модификаций и версий систем управления. Абстракция на уровне ActiveX маскирует способ интеграции в интерфейсе пользователя системы управления. Приложение может

Рис. 1. Ключевые технологии разработки программного обеспечения ЧПУ

быть использовано самостоятельно или интегрировано в систему ЧПУ в качестве отдельного подрежима.

XML DOM-технологии. Высокое быстродействие MS XML-компилятора делает его привлекательным при организации внутренней структуры данных приложений системы ЧПУ. Внутреннюю базу данных приложения реализуют на основе XML-DOM документа.

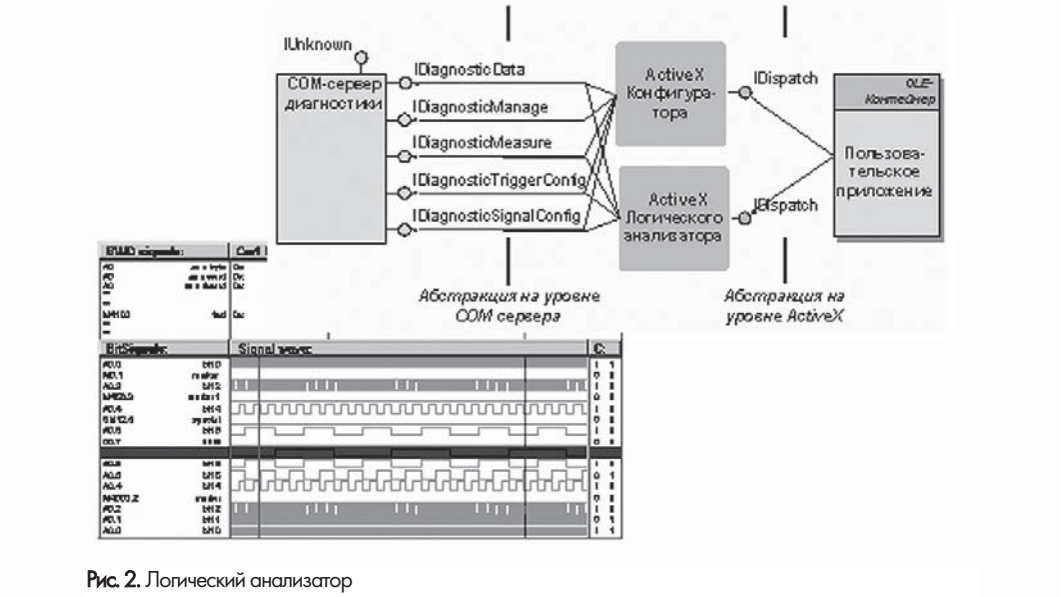

Рис. 2. Логический анализатор

Перспективные технологии разработки математического обеспечения систем управления

Практически каждые 1,5-2 года европейским разработчикам систем управления приходится осваивать новые технологии, менять компиляторы и среду разработки, переходить на новые программные платформы. Де-факто стандартом разработки ядра систем ЧПУ стал язык C++, и только в редких случаях остался язык C (рис. 3). При разработке терминальной части используют языки C++, VB, редко — Delphi, и довольно часто — смешанное решение, созданное на базе нескольких языков. В пользовательском интерфейсе, построенном на независимой платформе, стал очевиден отход от Java решения в пользу .NET, а доминирующим языком разработки стал язык C#.

С конца 90-х гг. прошлого века широко применяется компонентный (COM — Component Object Model) подход, позволяющий маскировать аппаратные особенности при создании программного обеспечения и реализовывать распараллеливание процесса разработки.

При организации машинных параметров систем управления нашла применение технология XML (eXtensible Markup Language). Ориентация на XML позволила использовать готовый программный инструментарий для работы с данными. В качестве иллюстрации можно привести пример системы ЧПУ, сконфигурированной с двумя каналами и восемью осями, которая имеет более 20 000 машинных параметров — и это не предел. Два года назад фирма Siemens презентовала на EMO систему ЧПУ с 64 осями, что вызывало непонимание специалистов из-за большого числа приводов, но таким образом фирма продемонстрировала возможности ядра системы. Современные системы обладают теоретически 12 каналами и 64 осями. Для работы с таким количеством данных применяются быстродействующие разборщики, как MSXML Parser, LibXML (Open Source) или разборщик, встроенный в .NET.

Технологию XSLT используют для генерации отчетов о состоянии машинных параметров или о сигналах проведенных диагностических измерений и настройках привода. С помощью XSLT также осуществляется конвертирование данных с одного формата в другой.

Сегодня Web-технологии используются для построения удаленного терминала системы управления. При этом с помощью скрипта и встроенного Internet-сервера в ядре генерируются экраны пользовательского интерфейса в стандартных браузерах.

Применение регулярных выражений в системах управления позволило унифицировать решение таких задач, как проверка на корректность и фильтрация пользовательского ввода значения, автоматическая генерация имен, поиск с использованием wildcards.

С помощью нечеткой логики и нейронных сетей сегодня решаются задачи адаптивного управления и прогнозирования надежности инструмента.

Этот небольшой список иллюстрирует практически неограниченные возможности применения технологий смежных областей, и именно здесь следует ожидать новые научные прорывы.

Пример реализации подсистемы диагностики инструмента в реальном времени

Задача сводится к прогнозированию износа инструмента в процессе обработки изделия с целью избежать повреждения дорогих изделий при поломке инструмента.

При этом нужно учитывать, что параметры качества помимо износа инструмента зависят от других повреждений в технологической системе, поэтому максимально допустимая величина износа [hз] может быть различной, в зависимости от того, какова доля его влияния на качество обработки.

Одноразовое определение K позволило бы рассчитать Tост до отказа инструмента лишь в том случае, если точки измеренных износов hз1 и hз2 располагались на прямой AB. В действительности, измеренные в реальных условиях работы данного инструмента, они относятся к неизвестной кривой, которая, как показано на рис. 4, колеблется при установившемся износе около прямой h = Kt. При расчете Ki на каждом временном отрезке Tзд производятся действия линейной интерполяции. Ki = tgαi на каждом отрезке TЗД будут иметь различные случайные значения, а Kср = ΣKi/i будет с ростом i приближаться к средней скорости нарастания износа работающего инструмента, которая определяется угловым коэффициентом K прямой h = Kt.

Следовательно, алгоритм прогнозирования $T_{ост}$ должен обеспечить действия по определению Ki и расчету $K_{ср}$ в течение всего периода стойкости инструмента. Тогда по мере приближения износа к [hз] достоверность прогноза $T_{ост}$ будет повышаться.

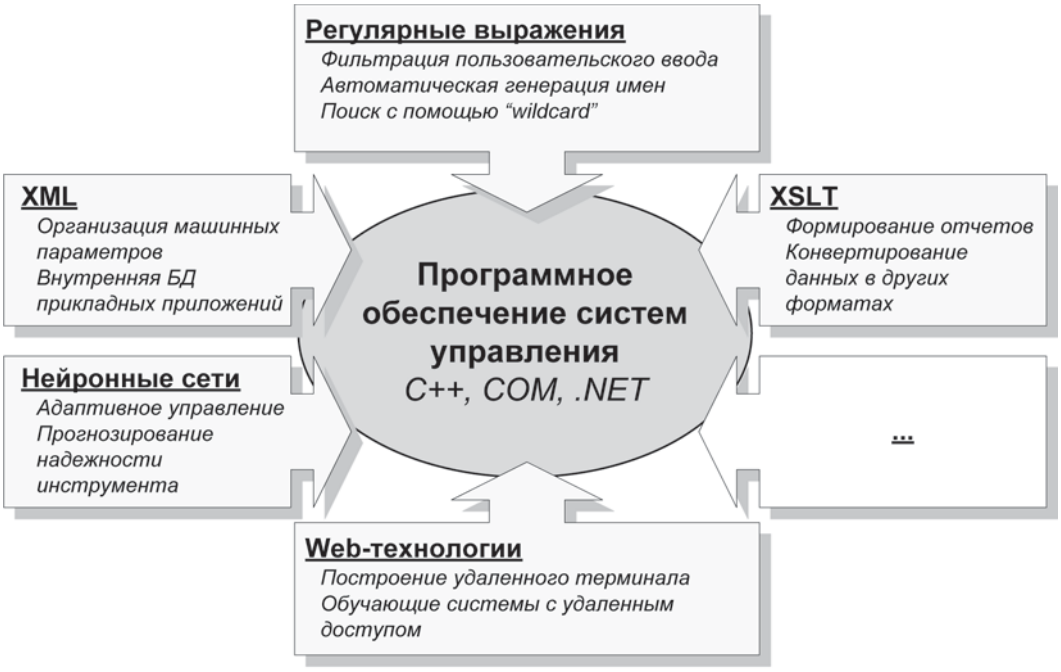

Рис. 3. Перспективные технологии разработки

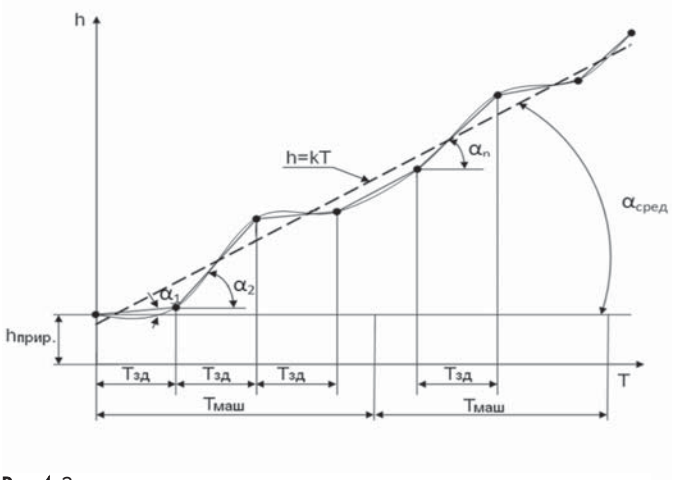

Рис. 4. Зависимость износ-время в стадии установившегося износа

В соответствии с описанной методикой, был разработан алгоритм прогнозирования остаточной стойкости.

Начальная инициализация алгоритма контроля инструмента подразумевает подготовительные операции по подключению к датчикам, а также получению данных от пользователя, к которым относятся:

— [h3] — максимальный допускаемый износ инструмента;

— $T_{ЗД}$ — задержка опроса датчиков, определяющая частоту получения информации;

— S — количество импульсов сигналов от датчиков силы в группе.

В работе использовались тензометрические датчики измерения составляющих силы резания как диагностических признаков износа.

Далее запускается цикл снятия информации с датчиков системы контроля инструмента с заданным ТЗД. Однако в процессе получения данных можно выделить несколько зон: холостого хода (смена заготовки, подвод инструмента), врезания, резания и выхода инструмента (рис. 5).

Для контроля износа инструмента должна использоваться только зона непосредственно резания. Учет данных из остальных зон не только не нужен, но и может привести к ошибке в прогнозировании остаточной стойкости инструмента. Для исключения данных из ненужных зон обработки вводятся алгоритмы определения момента начала процесса резания и определения момента окончания резания. Реализация данных этих алгоритмов идентична. Весь поток получаемых данных разбивается на группы из S последовательных импульсов сигнала (S задается на этапе начальной инициализации алгоритма контроля инструмента). Для каждой группы рассчитываются максимальное, минимальное и среднее значения. Далее сравниваются параметры последовательных групп и, в зависимости от их взаимного расположения, вырабатывается сигнал о начале или об окончании резания (рис. 6).

Таким образом, в случае, если получен сигнал о начале процесса врезания, а затем, после нескольких сигналов врезания, состояние стало неизменным, фиксируется момент установившегося процесса резания и начинается сбор данных для прогнозирования. Аналогично, в случае получения набора сигналов выхода инструмента, а затем их исчезновения, можно делать вывод о прекращении процесса обработки.

Однако получение сигнала может быть вызвано случайным выбросом данных, получаемых с датчика, а не реальным изменением состояния. С целью исключения подобных ситуаций, введен дополнительный уровень контроля сигналов. Последовательно рассматриваются 3-5 групп данных и, если сигнал в этих группах не повторяется (рис. 7), он считается ложным и игнорируется.

Все данные, получаемые в процессе холостого хода, не оказывают влияния на прогноз, но, тем не менее, собираются и анализи-

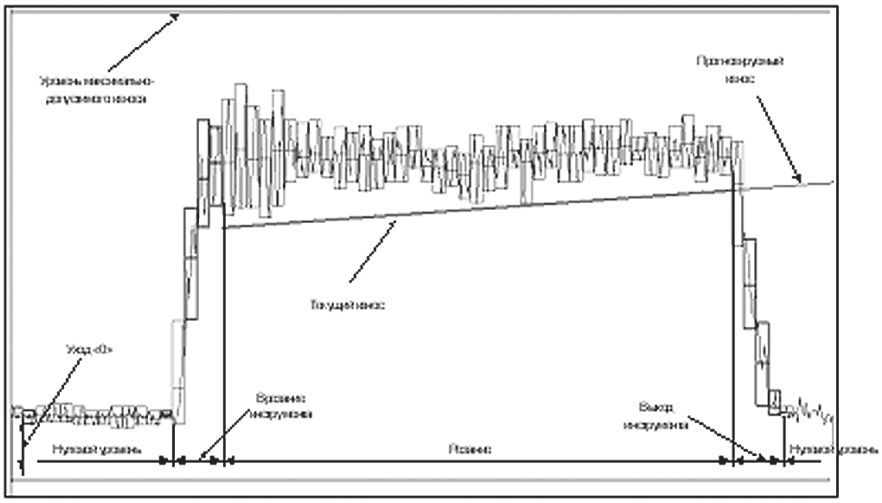

Рис.5. Виды сигналов датчика силы резания при обработке детали

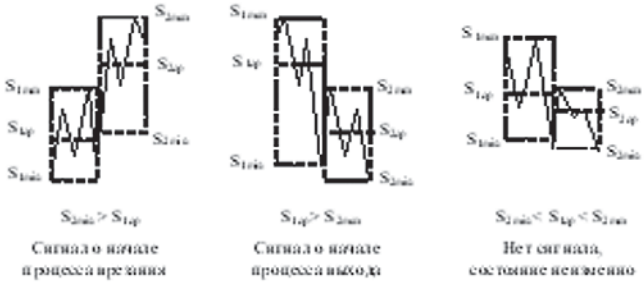

Рис. 6. Сигналы на изменение состояния процесса резания в зависимости от расположения групп данных

Рис. 7. Ложные сигналы смены состояния обработки

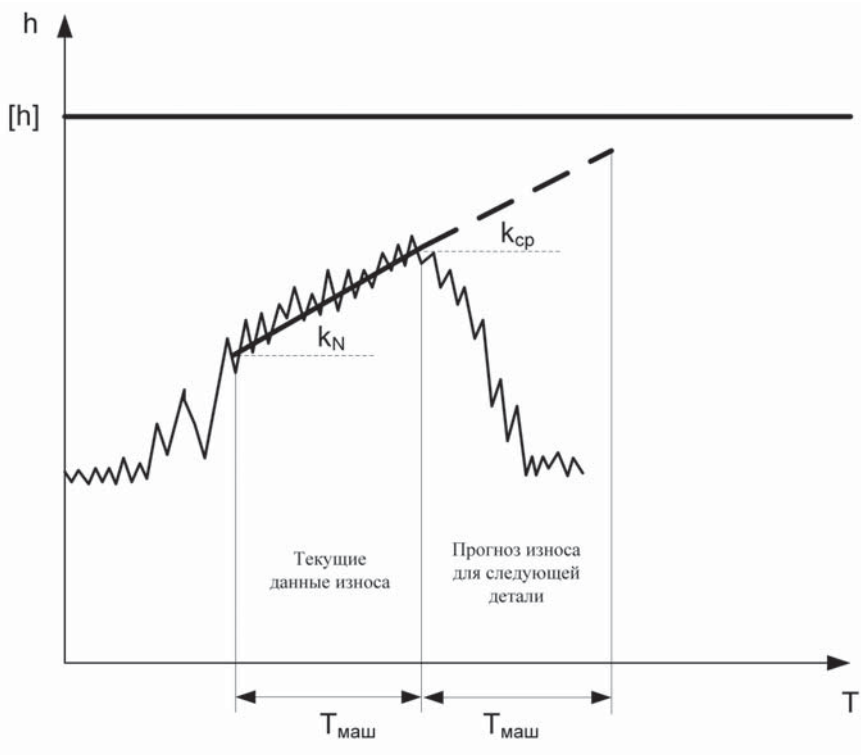

Рис. 8. Пример построения прогноза

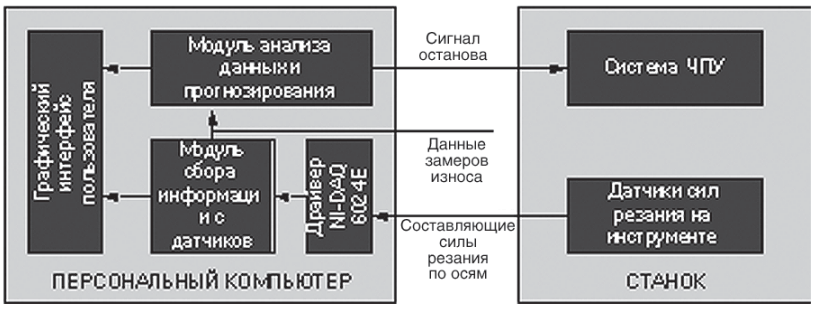

Рис. 9. Архитектура подсистемы прогнозирования остаточной стойкости инструмента

руются алгоритмом программной корректировки "нуля". Этот алгоритм вводится по той причине, что нулевой уровень сигнала может колебаться и для правильного анализа получаемых в процессе обработки значений это нужно учитывать. С этой целью рассчитывается среднее значение по всем данным, получаемым в процессе холостого хода до момента врезания, и полученная величина вычитается или прибавляется к значениям, получаемым в процессе резания.

Следует также учитывать, что данные, полученные в процессе обработки первой детали, необходимо исключить из работы алгоритма прогнозирования в связи с тем, что инструмент проходит приработку, и прогноз по первой детали может оказаться неверным.

Данные об износе инструмента, полученные в процессе обработки второй и последующих деталей, используются в алгоритме прогнозирования остаточной стойкости инструмента. Алгоритм рассчитывает коэффициент k наклона прямой по данным износа инструмента, полученным при обработке последней детали, находит среднее значение коэффициента наклона с учетом коэффициентов, полученных при обработке предыдущих деталей, и продолжает прямую износа с коэффициентом наклона $k_{ср}$ на расстояние, соответствующее $T_{МАШ}$ (рис.8).

Если отрезок полученной прямой прогноза для ТМАШ пересекает максимально допустимый уровень $[h_3]$, то выдается сигнал на прекращение процесса обработки.

Общая архитектура системы прогнозирования остаточной стойкости инструмента представлена на рис. 9.

Система прогнозирования включает: программные модули открытой системы ЧПУ, работающие на персональном компьютере под управлением ОС Windows XP, аппаратный модуль сопряжения с датчиками (платой ввода-вывода сигнала), датчики (силы, вибрации, акустической эмиссии и т.д.). Она может быть встроена в систему ЧПУ класса PCNC;

Данные об износе инструмента поступают с датчиков, расположенных на инструменте, в модуль сбора информации с датчиков. Из модуля сбора информации данные поступают в модуль анализа данных и отображаются графически в пользовательском интерфейсе программы. В свою очередь, модуль анализа данных отображает информацию о прогнозе в пользовательском интерфейсе и, в случае отрицательного прогноза, выдает сигнал на систему ЧПУ.

В качестве средства разработки программных модулей используется среда разработки Microsoft Visual Studio 2005 и API функции, поставляемые с платой National Instruments DAQ 6024E.

Заключение

Изложенные выше идеи позволили за короткие сроки реализовать подсистему диагностики инструмента с применением компонентного подхода, а также решить задачу получения с датчиков и обработки больших объемов данных в реальном времени.