УДК 004.415.2.031.43

В.В. Никифоров, В.И. Шкиртиль

ОЦЕНКА ВРЕМЕНИ ОТКЛИКА ЦЕПОЧЕК ЗАДАЧ В РАСПРЕДЕЛЕННЫХ СИСТЕМАХ РЕАЛЬНОГО ВРЕМЕНИ

Никифоров В.В., Шкиртиль В.И. **Оценка времени отклика цепочек задач в распределенных системах реального времени.**

Аннотация. Представлен метод оценки продолжительности обработки внешних событий программными приложениями в распределенных системах реального времени. Показано, что на выполнимость таких приложений могут влиять вариации продолжительности исполнения отдельных задач и передачи отдельных сообщений. Такое влияние необходимо учитывать при оценке выполнимости цепочек задач, участвующих в реакции на конкретные типы внешних событий.

Ключевые слова: системы реального времени, модели многозадачных программных систем, распределенные вычисления, динамическая корректность программных приложений.

Nikiforov V.V., Shkirtil V.I. **Response time estimation for task chains in destributed real-time systems.**

Abstract. A method for estimation of time, requiered for external events processing by distributed program applications is presented. It is shown that feasibility of such applications may depend on variation of task execution time and message delivery time. The influence of this variation shall be taken into account during feasibility checking for task chain, that participate in reaction to concrete separate external events.

Keywords: real-time systems, models of multitask software systems, distributed computations, dynamic correctness of software applications.

1. Введение. Ключевое свойство программных приложений для систем реального времени (СРВ) состоит в том, что они работают «в структуре времени, определяемой ходом внешних процессов». Такое соответствие между ходом исполнения программных компонент СРВ и ходом внешних процессов обусловлено наличием более или менее жестких ограничений на временные рамки для информационных обменов с внешними процессами. Для анализа программных приложений СРВ на предмет соблюдения подобных ограничений строятся соответствующие вычислительные модели.

В последние сорок лет технология построения и анализа вычислительных моделей СРВ формировалась в основном для однопроцессорных систем, в которых реакция системы на внешнее событие состоит в исполнении одной конкретной задачи [1–7]. Для программных систем со статическими приоритетами задач базовый метод оценки

выполнимости опирается на вычисление времени отклика R_k каждой из задач τ_k. Время отклика вычисляется как сумма факторов:

$$R_k = C_k + I_k + B_k + J_k, \quad (1)$$

где C_k (фактор веса) — максимальный объем процессорного времени, требуемого для выполнения τ_i; I_i (фактор приоритета) — максимальная продолжительность ожидания момента освобождения процессора, занятого более приоритетными задачами; B_i (фактор блокирования) — максимальная продолжительность ожидания ресурсов, занимаемых менее приоритетными задачами; J_i (фактор дребезга) — максимальная задержка регистрации задачи τ_i.

Ниже рассмотрены базовые принципы построения и анализа моделей, ориентированные на проверку выполнимости программных приложений для распределенных СРВ, содержащих ряд процессоров, соединенных в локальную вычислительную сеть.

2. Статическое разбиение задач и ресурсов по узлам сети. Для большинства жестких распределенных СРВ с жесткими требованиями реального времени множество прикладных задач привязывается к узлам сети статически. То есть, имеет место следующее ограничение.

Ограничение 1. Множество прикладных задач статически разбивается на подмножества, распределяемые между процессорами, составляющими распределенную систему [2].

Ниже рассматриваются вычислительные модели программных приложений распределенных СРВ, отвечающих ограничению 1. Это означает, что вычислительная модель программного приложения распределенной СРВ должна включать перечень $\{P_1,...,P_m\}$ процессоров (узлов), реализующих функции распределенной системы.

Определение 1. В рамках рассматриваемой вычислительной модели программного приложения распределенной СРВ $i^{\text{ый}}$ узел системы представляет собой процессор P_i, обеспечивающий

— исполнение подмножества $\{\tau_{i,1},...\tau_{i,n_i}\}$ задач, входящих в моделируемое программное приложение,

— доступ задач $\{\tau_{i,1},...\tau_{i,n_i}\}$ к подмножеству ресурсов $\{g_{i,1},...,g_{i,m_i}\}$, контролируемых этим процессором.

В этом смысле процессор P_i владеет ресурсами $\{g_{i,1},...,g_{i,m_i}\}$ и задачами $\{\tau_{i,1},...\tau_{i,x}\}$. Связь задач, принадлежащих процессору $i^{\text{го}}$ узла моделируемой распределенной СРВ с задачами, принадлежащими другим узлам, обеспечивается средствами коммуникационной подсистемы.

2.1. Отношения предшествования. Еще одна особенность, отличающая распределенные СРВ, состоит в необходимости усложнения вычислительных моделей в части, касающейся установления соответствий между типами внешних событий и задачами, участвующими в реакции на эти события. Для однопроцессорных систем может оказаться достаточным указание однозначного соответствия между типами внешних событий и прикладными задачами, исполняемыми в ответ на возникновение таких событий (очередное внешнее событие определенного типа обрабатывается очередным экземпляром соответствующей задачи). Непременным свойством распределенных СРВ является наличие таких внешних событий, которые обрабатываются задачами, размещаемыми в различных узлах сети (иначе это не распределенная система, а разрозненное множество несвязанных узлов).

Задачи, совместно реагирующие на внешние события определенного типа, связываются *отношениями предшествования*. Если обработка внешнего события $i^{\text{го}}$ типа состоит из ряда этапов $s_{i,1}$, $s_{i,2}$, . . ., и каждый из этапов реализуется отдельной задачей (символ $s_{i,k}$ играет роль ссылки на эту задачу), то некоторые пары этих этапов (соответственно, некоторые пары задач) связываются отношениями предшествования. Наличие отношения предшествования $s_{i,k} \to s_{i,l}$ означает, что задача, реализующая этап $s_{i,l}$, начинает исполняться только после того, как полностью завершается исполнение задачи, реализующей этап $s_{i,k}$.

Структура ориентированного графа, представляющего совокупность тех отношений предшествования, которые связывают задачи, участвующие в обработке внешних событий *i*-го типа, может быть более или менее сложной. В самом общем случае это может быть произвольный ориентированный граф без контуров. Отметим, что отсутствие контуров означает, в частности, что две различные ссылки $s_{i,l}$ и $s_{i,k}$ на задачи, участвующие в обработке событий *i*-го типа, соответствуют различным задачам (то есть, в ответ на возникновение внешне-

го события *i*-го типа каждая задача, участвующая в обработке этого события, активизируется лишь единожды).

Различными исследователями рассматривались модели с графами предшествования в виде дерева или леса [2, 4]. Такая структура отношений предшествования может использоваться и в программных приложениях для однопроцессорных систем [8].

Для большинства СРВ достаточно использовать вычислительные модели с теми или иными ограничениями на сложность структуры графов предшествования. Практика построения СРВ показывает, что в большинстве случаев множество отношений предшествования имеет простую структуру и представимо в виде цепочек (каждому типу внешних событий соответствует цепочка задач).

Определение 2. Цепочка Ψ_i задач — это последовательность задач, связанных отношениями предшествования. Цепочка Ψ_i задается списком ссылок $\{s_{i,1}, s_{i,2},..., s_{i,l_i}\}$ на составляющие ее задачи: задача, соответствующая ссылке $s_{i,k}$, является предшествующей по отношению к задаче, которая соответствует ссылке $s_{i,k+1}$. Условием активизации последующей задачи является завершение исполнения ее предшественницы. Условием активизации задачи, соответствующей ссылке $s_{i,1}$ (корневой задачи цепочки), является внешнее событие, реакция на которое заключается в исполнении последовательности задач, составляющих цепочку Ψ_i. Отношение $\tau_a \in \Psi_i$ означает, что на задачу τ_a есть ссылка из цепочки Ψ_i. Тип внешних событий, активизирующих корневую задачу цепочки Ψ_i, обозначим символом E_i.

Ниже рассматриваются программные приложения, отвечающие следующему ограничению.

Ограничение 2. Отношения предшествования задач, составляющих программное приложение, задаются непересекающимися цепочками задач.

Под пересечением двух цепочек Ψ_i и Ψ_j понимается существование такой задачи τ_a, на которую имеются ссылки как из цепочки Ψ_i, так и из цепочки Ψ_j ($\tau_a \in \Psi_i$ и $\tau_a \in \Psi_j$). При выполнении ограничения 2 такие пересечения исключены.

Ограничение 2 означает, что имеет место взаимнооднозначное соответствие между ссылками и задачами. Символ τ с определенными индексами указывает принадлежность задачи определенному узлу

распределенного программного приложения и уровень приоритета этой задачи; символ s с определенными индексами указывает принадлежность задачи определенной цепочке и ее порядковый номер в цепочке. В дальнейшем вместо оборота «задача, представляемая ссылкой s», используется оборот «задача s».

Если в однопроцессорной системе отношения предшествования отвечают ограничению 2, то каждую цепочку Ψ_i целесообразно представлять единой задачей τ_i. При этом отношение предшествования реализуется порядком следования соответствующих блоков в теле реализующей τ_i последовательной программы (для распределенного программного приложения подобная замена неосуществима, поскольку задачи, принадлежащие одной и той же цепочке, принадлежат различным процессорам).

Аналогично, если в распределенной системе все задачи, соответствующие фрагменту $s_{i,k},...,s_{i,k+l_i}$ цепочки Ψ_i, принадлежат одному и тому же процессору, то все эти задачи целесообразно представлять единой задачей и в определении цепочки Ψ_i заменить весь фрагмент ссылкой на такую единую задачу. В этой связи будем считать, что рассматриваемые вычислительные модели программных приложений распределенных СРВ отвечают следующему ограничению.

Ограничение 3. В каждой цепочке Ψ_i задачи, соответствующие двум соседним ссылкам $s_{i,k}$ и $s_{i,k+1}$, принадлежат различным процессорам.

Рассматриваемые далее вычислительные модели для распределенных СРВ должны обеспечивать установление соответствий между типами внешних событий и списками прикладных задач распределенного программного приложения. Каждому типу E_i внешних событий ставится в соответствие список Ψ_i тех задач (реализуемых, возможно, различными процессорами), выполнение которых обеспечивает должную реакцию системы на возникновение события этого типа.

2.2. Локальные и удаленные ресурсы. В соответствии с определением 1 каждый ресурс принадлежит конкретному процессору. Принадлежность ресурса g процессору P означает, что исполнение критических секций по доступу к ресурсу g осуществляется процессором P. В соответствии с рассматриваемой вычислительной моделью программного приложения распределенной СРВ процессор P,

владеющий задачей τ, называется *локальным* процессором задачи τ (задача τ принадлежит своему локальному процессору). Ресурс g, принадлежащий локальному процессору задачи τ, является по отношению к задаче τ *локальным ресурсом*. Если ресурс g, используемый задачей τ, не является ее локальным ресурсом, то g называется *удаленным* ресурсом задачи τ.

Из определения 1 следует, что в ряду критических интервалов по доступу к информационным ресурсам в задаче $\tau_{i,k}$ (задачи, принадлежащей процессору P_i) могут быть только критические интервалы по доступу к локальным ресурсам процессора P_i. Критические интервалы по доступу к удаленным ресурсам находятся в кодах задач, принадлежащих другим процессорам моделируемой системы.

Код отдельной задачи $\tau_{i,k}$, исполняемой процессором P_i, может содержать несколько критические интервалов по доступу к различным локальным ресурсам. Допускаются пересечения (вложенность, сцепление) этих критических интервалов. Определение максимальной продолжительности исполнения для таких задач может осуществляться, например, методами, представленными в [2, 9–12].

Если при выполнении ограничения 3 длина последовательности задач в цепочке Ψ_i равна l_i, то для обработки каждого из внешних событий типа E_i требуется $l_i - 1$ раз использовать услуги коммуникационной системы, связывающей узлы распределенной СРВ. В частности, если все информационные ресурсы, требуемые для обработки внешних событий типа E_i, принадлежат одному и тому же процессору, то цепочка Ψ_i вырождена (содержит единственную задачу — корневую задачу, соответствующую ссылке $s_{i,1}$). Ресурсы, удаленные по отношению к задаче, представляемой ссылкой $s_{i,1}$, для обработки внешних событий типа E_i не требуются. Следовательно, для обработки таких событий не требуется использование коммуникационной подсистемы. Тогда проверка гарантий своевременности исполнения Ψ_i может осуществляться методами, разработанными для однопроцессорных систем. Если обработка событий типа E_i требует обращения к информационным и специальным исполнительным ресурсам, принадлежащим различным узлам распределенной системы, то l_i строго больше единицы.

3. Модели программных приложений в распределенных СРВ. В рамках представленного подхода вычислительная модель программного приложения распределенной СРВ должна содержать:

— перечень процессоров (узлов, модулей), составляющих моделируемую распределенную систему,

— указание для каждого из процессоров P_i состава принадлежащих ему задач $\tau_{i,1},...\tau_{i,n_i}$ и информационных ресурсов $g_{i,1},...,g_{i,m_i}$,

— перечень типов внешних событий с указанием для каждого из них соответствующей цепочки Ψ_i, той последовательности задач, которая обеспечивает должную реакцию программного приложения распределенной СРВ на возникновение очередного внешнего события типа E_i,

— указание для каждой цепочки Ψ_i последовательности $\{s_{i,1}, s_{i,2},..., s_{i,l_i}\}$ ссылок на задачи, исполняемые в ответ на возникновение внешних событий типа E_i.

Для каждой задачи $s_{i,k}$ задаются параметры, касающиеся использования ею исполнительных и информационных ресурсов:

— максимальный объем процессорного времени $C_{i,k}$, требуемый для ее выполнения,

— максимальная продолжительность $C_{i,k}(g_{i,l})$ исполнения критического интервала по доступу к ресурсу $g_{i,l}$.

Для каждого типа E_i внешних событий задается значение T_i периода их следования — минимально возможной продолжительности интервала времени между соседними экземплярами событий этого типа.

В силу ограничения 3 каждой паре задач, связанных отношением предшествования, должен соответствовать определенный тип сообщений, передаваемых коммуникационной подсистемой из узла, владеющего предшествующей задачей в узел, владеющий последующей задачей. Необходимо указывать допустимый объем каждого такого типа сообщений.

3. Исполнение цепочек задач в распределенных СРВ. Прикладные программные комплексы, отвечающие вычислительным моделям с ограничениями 1–3, реагируют на внешние события типа E_i исполнением цепочки задач Ψ_i. Входящие в состав Ψ_i задачи размещаются

в узлах распределенной системы, обмениваясь сообщениями, передаваемыми средствами коммуникационной подсистемы. Методы оценки продолжительности передачи элементарных сообщений и пакетов сообщений для локальных вычислительных сетей с механизмами арбитража (например, с интерфейсами типа CAN) аналогичны методам оценки времени отклика задач в однопроцессорных СРВ [13,14].

Для представленных ниже методов оценки продолжительности исполнения цепочек задач наряду с ограничениями 1–3 принимаются следующие дополнительные ограничения.

Ограничение 4. Приоритеты задач, входящих в состав распределенного программного приложения, фиксированы (назначаются статически). Задачи $\tau_{i,k} \in P_i$ и $\tau_{i,l} \in P_i$, принадлежащие одному и тому же процессору, имеют различные значения приоритетов — $\text{prio}(\tau_{i,k}) \neq \text{prio}(\tau_{i,l})$.

Ограничение 5. Состав сообщений, передаваемых между узлами коммуникационной системы, ограничивается сообщениями, связывающими соседние задачи в цепочках Ψ_i, образующих распределенное программное приложение. Приоритеты сообщений, адресуемых различным задачам, различны.

3.1. Составляющие интервала исполнения цепочек задач. Исполнение цепочки Ψ_i начинается в момент возникновения внешнего события, активизирующего корневую задачу цепочки Ψ_i, которая становится текущей задачей очередного исполнения этой цепочки. Далее циклически исполняются следующие шаги:

Шаг 1. Локальный процессор исполняет текущую задачу совместно с другими принадлежащими ему задачами в соответствии с принятой дисциплиной планирования. Если текущая задача является замыкающей задачей цепочки Ψ_i, то с ее завершением завершается исполнение цепочки, иначе выполняется шаг 2.

Шаг 2. Перед завершением текущей задачи цепочки Ψ_i ею формируется элементарное сообщение (или пакет элементарных сообщений), направляемое очередной задаче цепочки Ψ_i. В силу ограничения 3 организуется передача этого сообщения на другой узел сети.

Шаг 3.Коммуникационная подсистема реализует передачу сформированного сообщения совместно с передачей сообщений, формируемых в рамках исполнения других активных цепочек. Узел, принявший сообщение, активизирует очередную задачу цепочки Ψ_i. Вновь

активизированная задача становится текущей задачей этой цепочки. Выполняется переход к очередному исполнению шага 1.

Для систем со статическими приоритетами задач продолжительность исполнения каждой задачи (суммарная продолжительность шагов 1 и 2) определяется методами, представленными в главе 2 с учетом:

— значений параметров задачи,
— используемой дисциплины планирования,
— принятыми способами назначения приоритетов задач,
— принятыми протоколами доступа к разделяемым ресурсам.

Продолжительность передачи сообщений (шаг 3) по локальной сети, построенной на базе интерфейса с механизмом арбитража (зависит от объема передаваемых сообщений, в частности, значений их приоритетов, рабочей частоты сети), определяется методами, представленными в [13,14].

Временные ограничения, непосредственно вытекающие из высокоуровневых спецификаций, требований к распределенным системам, представляются, как правило, ограничениями продолжительности реакции системы на каждый тип внешних событий. Это означает, что для распределенных систем естественно накладывать ограничения лишь на общую продолжительность интервалов исполнения каждой из цепочек задач (то есть, на значение времени отклика R_i каждой из цепочек Ψ_i). Для каждой цепочки Ψ_i указывается *относительный предельный срок* D_i выполнения всей цепочки, то есть, допустимое значение длины временного интервала между возникновением очередного внешнего события типа E_i и моментом завершения соответствующего исполнения заключительной задачи цепочки Ψ_i. Ограничений на время отклика промежуточных задач цепочки Ψ_i при этом не накладывается.

В таких условиях распределенное программное приложение при выбранной дисциплине планирования следует считать выполнимым, если для каждой цепочки справедливо неравенство

$$R_i \le D_i . \qquad (2)$$

Актуален выбор такой дисциплины планирования, такого способа назначения приоритетов задач и сообщений, который позволил бы добиться выполнимости распределенного приложения. Актуально также

отыскание метода проверки выполнения неравенств (11), то есть, отыскание метода оценки значений R_i.

3.2. Приоритеты задач и сообщений. Дисциплина планирования определяет порядок предоставления исполнительных ресурсов тем объектам, которые претендуют на их использование (заданиям предоставляются процессоры, сообщениям — ресурсы коммуникационной подсистемы).

В тесно связанной компьютерной системе с несколькими исполнительными ресурсами (симметричными процессорами, многоядерными процессорами) текущий глобальный статус компонент системы и информация о текущей загрузке всех процессоров (ядер) может быть доступна ценой небольших издержек. Такая система может использовать централизованного диспетчера/планировщика. Если же каждый процессор имеет собственного планировщика, то решения и действия всех планировщиков выполняются когерентно [2]. Иная ситуация имеет место в таких слабо связанных системах, какими являются распределенные СРВ: оказывается слишком дорого хранить глобальную информацию о текущей загрузке процессоров и текущих состояниях компонент системы.

В рамках принятых выше ограничений планировщики различных процессоров выполняют действия по планированию исполнения задач и контролю доступа к информационным ресурсам независимо. Оптимальная дисциплина назначения приоритетов задач неизвестна. Неизвестна и оптимальная дисциплина назначения приоритетов сообщений, связывающих различные задачи в каждой из цепочек Ψ_i. Неизвестно также, существуют ли такие оптимальные дисциплины планирования.

В таких условиях остается искать так или иначе обоснованные эвристические дисциплины планирования. Для распределенных систем представляется естественным использование эвристической дисциплины планирования задач, которая аналогична оптимальной RM дисциплине планирования в однопроцессорных системах.

Определение 3. Темпо-монотонная дисциплина планирования для распределенных программных приложений (D-RM дисциплина планирования) предполагает статическое назначение приоритетов цепочек. Приоритеты цепочек $\mathrm{prio}(\Psi_i)$ снижаются в порядке возрастания значений T_i — значений периодов следования внешних событий E_i, активизирующих корневую задачу цепочки. Приоритет корневой задачи цепочки равен $\mathrm{prio}(\Psi_i)$ — приоритету самой цепочки. Приоритеты

остальных задач цепочки Ψ_i монотонно понижаются — чем дальше задача отстоит от начала цепочки Ψ_i, тем ниже ее приоритет.

Для того, чтобы гарантированно выполнялось ограничение 4, дополним определение D-RM дисциплины планирования следующими двумя уточнениями.

Уточнение 1. Если для двух типов внешних событий E_i и E_j имеет место равенство $T_i = T_j$ (значения периодов совпадают), то для соответствующих им цепочек Ψ_i и Ψ_j устанавливаются различные значения приоритетов — $\text{prio}(\Psi_i) \neq \text{prio}(\Psi_j)$. Решение о том, какой из цепочек присваивается более высокий приоритет, может определяться какими-то дополнительными соображениями.

Уточнение 2. Если $\text{prio}(\Psi_i) \succ \text{prio}(\Psi_j)$ (приоритет цепочки Ψ_i выше, чем приоритет цепочки Ψ_j), то приоритет любой задачи из цепочки Ψ_i выше приоритета любой задачи из цепочки Ψ_j.

В случае распределенной СРВ дисциплина планирования может считаться полностью заданной в том случае, если наряду с указанием приоритетов задач указываются и приоритеты сообщений, передаваемых коммуникационной подсистемой. Поэтому вводим еще одно уточнение.

Уточнение 3. D-RM дисциплина планирования определяет способ назначения приоритетов для всех типов сообщений, передаваемых коммуникационной подсистемой: приоритет сообщения, передаваемого через коммуникационную систему задачей $\tau_{i,k}$, равен приоритету самой задачи $\tau_{i,k}$.

4. Оценка выполнимости цепочек задач. Совместное использование методов оценки времени отклика для задач в однопроцессорных СРВ и предложенного в [13,14] подхода к оценке времени доставки сообщений в локальных сетях позволяет для каждой цепочки задач Ψ_i оценить значение R_i ее времени отклика.

Ниже представлен подход к оценке R_i в приложениях, для которых выполняется следующее ограничение.

Ограничение 6. Время отклика R_i каждой цепочки задач Ψ_i не превосходит значения периода T_i следования внешних событий E_i, активизирующих корневую задачу цепочки Ψ_i.

Для распределенных СРВ не известен универсальный способ построения критического сценария внешних событий для каждой цепочки задач Ψ_i. В этих условиях универсальный метод оценки выполнимости может строиться на основе приближенных оценок $R_i^* \geq R_i$ для значений времени отклика цепочек Ψ_i. Выполнимость распределенного приложения гарантируется, если при любом i справедливо неравенство $R_i^* \leq D_i$.

В случае использования приближенной оценки R_i^* вместо точного значения R_i такая гарантия, возможно, обеспечивается за счет избыточной производительности исполнительных ресурсов распределенной системы.

4.1. Оценка времени отклика высокоприоритетной цепочки задач. Рассмотрим составляющие времени отклика для цепочки Ψ_1:

$$R(\Psi_1) = \sum_{1 \leq i \leq l_1} R(s_{1,i}) + \sum_{1 \leq i \leq l_1 - 1} R(m_{1,i}), \tag{3}$$

где l_1 — число задач в цепочке Ψ_1; $R(s_{1,i})$ — продолжительность интервала времени исполнения задачи $s_{1,i}$ в рамках критического сценария; $R(m_{1,i})$ — продолжительность интервала времени доставки сообщения, направляемого от задачи $s_{1,i}$ в адрес задачи $s_{1,i+1}$ в рамках критического сценария.

Величина $R(s_{1,i})$ равна сумме

$$R(s_{1,i}) = C(s_{1,i}) + B(s_{1,i}). \tag{4}$$

Здесь $C(s_{1,i})$ — фактор веса (объем процессорного времени, требуемого для выполнения задачи $s_{1,i}$ в рамках критического сценария). Второе слагаемое $B(s_{1,i})$ — фактор блокирования (продолжительность в рамках критического сценария блокирования доступа задачи $s_{1,i} \in P$ к тем разделяемым информационным ресурсам, которые контролируются процессором P). Величина $B(s_{1,i})$ заведомо ограничена значением

$$B(s_{1,i}) \leq B^*(s_{1,i}), \tag{5}$$

где $B^*(s_{1,i})$ вычисляется как фактор блокирования задачи $s_{1,i}$ со стороны остальных задач, принадлежащих процессору P по правилам, определенным для однопроцессорных систем (см. [12]).

Отметим, что в выражении (4) отсутствует фактор приоритета: в рамках рассматриваемой вычислительной модели задачи из цепочки Ψ_1 не могут вытесняться, поскольку приоритеты любой задачи из Ψ_1 выше приоритетов любой задачи из остальных цепочек распределенного программного приложения и в силу ограничения 6 интервалы существования однотипных заданий не пересекаются.

Величина $R(m_{1,i})$ в выражении (3) равна сумме:

$$R(m_{1,i}) = L(m_{1,i}) + B(m_{1,i}), \qquad (6)$$

где $L(m_{1,i})$ — фактор веса, продолжительность интервала передачи сообщения, направляемого в рамках критического сценария задачей $s_{1,i}$ задаче $s_{1,i+1}$. Максимально возможный вес $L^*(m_{1,i})$ интервала передачи сообщения типа $m_{1,i}$ определяется параметрами вычислительной модели. Второе слагаемое $B(m_{1,i})$ — фактор блокирования, продолжительность (в рамках критического сценария) ожидания узлом, владеющим задачей $s_{1,i}$, момента освобождения коммуникационного канала, занятого пересылкой менее приоритетного сообщения. Максимально возможное значение $B^*(m_{1,i})$ одинаково для любых сообщений из цепочки Ψ_1 (а также из и любой другой цепочки задач), оно равно максимальной продолжительности L_{elem} передачи элементарного сообщения по коммуникационной линии.

Отметим, что фактор приоритета в (6) отсутствует, поскольку в рамках рассматриваемой вычислительной модели приоритет любого сообщения из цепочки Ψ_1 выше приоритета любого сообщения из остальных цепочек распределенного программного приложения, а также в силу ограничения 6 (интервалы передачи однотипных сообщений не пересекаются).

В общем случае для точного определения значения R_1 требуется искать критический сценарий для цепочки Ψ_1 путем перебора всевозможных сценариев внешних событий. Если такой перебор требует чрезмерно больших вычислительных ресурсов, остается довольство-

ваться приближенной оценкой R_1^* для значения R_1 времени отклика цепочки Ψ_1:

$$R_1 \le R_1^* = \sum_{1 \le i \le l_1} (C^*(s_{1,i}) + B^*(s_{1,i})) + \sum_{1 \le i \le l_1 - 1} L^*(m_{1,i}) + B^*(m_{1,i}) . \qquad (7)$$

При $R_1^* \le D_1$ своевременность исполнения цепочки Ψ_1 гарантирована для любых допустимых сценариев следования внешних событий.

4.2. Фактор сдвига. Активизация корневой задачи каждой цепочки жестко привязана к моментам возникновения соответствующих внешних событий. Параметр T_i, определяющий минимальный интервал времени между активизациями задачи $s_{i,1}$, характеризует интенсивность использования исполнительных ресурсов распределенной системы цепочкой Ψ_i. Периодичность активизации корневых задач точно соответствует периодичности возникновения соответствующих им внешних событий. Время отклика $R(s_{i,1})$ корневых задач определяется в точном соответствии с методами, определенными для однопроцессорных систем. При этом минимальный интервал времени $R(s_{i,1})$ между порождениями заданий типа $s_{i,1}$ в точности равен T_i.

Активизация некорневых задач $s_{i,k}$ ($k > 1$) связана с возникновением внешних событий типа E_i опосредованно, через исполнение предшествующих задач цепочки Ψ_i и передачу соответствующих сообщений. При $k > 1$ продолжительность интервалов времени между порождениями заданий типа $s_{i,k}$ варьируется в более широких пределах по сравнению с заданиями типа $s_{i,1}$, поскольку зависит от вариаций продолжительности исполнения предшествующих задач цепочки Ψ_i и вариаций продолжительности передачи сообщений, генерируемых этими задачами. Поэтому, несмотря на косвенную связь между возникновением событий типа E_i и активизацией некорневой задачи $s_{i,k}$, минимальный интервал времени между порождениями заданий типа $s_{i,k}$ может оказаться существенно меньшим, чем T_i.

Вариации продолжительности исполнения фрагментов цепочки Ψ_i приводят к сдвигам моментов порождения задач $s_{i,k}$ и сообщений

$m_{i,k}$, следующих за такими фрагментами. Подобные сдвиги влияют на значения максимально возможной продолжительности исполнения фрагментов менее приоритетных цепочек Ψ_j ($i < j$) аналогично тому, как фактор дребезга (1) влияет на время отклика задач в однопроцессорных системах. Назовем такое влияние фактором сдвига.

Фактор сдвига $S(s_{i,k})$ для некорневой задачи $s_{i,k}$ определяется выражением

$$S(s_{i,k}) = R_{\max}(...,s_{i,k}) - R_{\min}(...,s_{i,k}), \quad (8)$$

где $R_{\max}(...,s_{i,k})$ и $R_{\min}(...,s_{i,k})$ соответственно максимальная и минимальная продолжительность интервала исполнения фрагмента цепочки Ψ_i, предшествующего исполнению задачи $s_{i,k}$.

Эффект сдвига следует учитывать и при оценке продолжительности передачи сообщений. Фактор сдвига $S(m_{i,k})$ при передаче сообщения $m_{i,k}$ определяется выражением

$$S(m_{i,k}) = R_{\max}(...,m_{i,k}) - R_{\min}(...,m_{i,k}), \quad (9)$$

где $R_{\max}(...,m_{i,k})$ и $R_{\min}(...,m_{i,k})$ соответственно максимальная и минимальная продолжительность исполнения фрагмента цепочки Ψ_i, предшествующего передаче сообщения $m_{i,k}$.

Поскольку критический сценарий неизвестен, нет возможности определить точные значения параметров $R_{\max}(...,s_{i,k})$, $R_{\min}(...,s_{i,k})$, $R_{\max}(...,m_{i,k})$ и $R_{\min}(...,m_{i,k})$. В таких условиях приходится ограничиться оценками этих параметров

$$R^*_{\max}(...,s_{i,k}) \geq R_{\max}(...,s_{i,k}), \quad R^*_{\min}(...,s_{i,k}) \leq R_{\min}(...,s_{i,k}),$$

$$R^*_{\max}(...,m_{i,k}) \geq R_{\max}(...,m_{i,k}), \; R^*_{\min}(...,m_{i,k}) \leq R_{\min}(...,m_{i,k}).$$

При использовании приведенных приближенных оценок вместо точных значений в формулах (8) и (9) получим приближенные оценки $S^*(s_{i,k})$ и $S^*(m_{i,k})$ значений факторов сдвига. Приближенные оценки значений факторов сдвига используются в вычислениях оценок R_i^* времени отклика для цепочек задач при $i > 1$. Из соображений удобства изложения такие цепочки будем называть низкоприоритетными цепочками (приоритеты этих цепочек ниже приоритета цепочки Ψ_1).

Факторы сдвига $S(s_{i,k})$ и $S(m_{i,k})$ влияют на значение времени отклика R_j всех менее приоритетных (относительно Ψ_i) цепочек задач. В частности, $S(s_{1,k})$, $S(m_{1,k})$ влияют на время отклика R_j каждой из низкоприоритетных цепочек. Как отмечалось выше, активизация корневой задачи каждой цепочки (в том числе и задачи $s_{1,1}$) жестко привязана к моментам возникновения внешних событий, обрабатываемых цепочкой. Следовательно, $S(s_{1,1}) = 0$. Вариация продолжительности исполнения задачи $s_{1,1}$ равна значению фактора сдвига $S(m_{1,1})$ передачи сообщения $m_{1,1}$:

$$S(m_{1,1}) \le R_{\max}(s_{1,1}) - R_{\min}(s_{1,1}). \qquad (10)$$

Величина $R_{\max}(s_{1,1})$ заведомо не превосходит суммы двух слагаемых — фактора веса $C_{1,1}$ и оценки фактора блокирования $B^*(s_{1,1})$ задачи $s_{1,1}$:

$$R_{\max}(s_{1,1}) \le C_{1,1} + B^*(s_{1,1}) = R^*{}_{\max}(s_{1,1}), \qquad (11)$$

(фактор приоритета отсутствует, поскольку узел, владеющий задачей $s_{1,1}$, не содержит более приоритетных задач).

Неравенство (11) превращается в равенство только в том случае, если существует сценарий событий, при котором порождается такой экземпляр задачи $s_{1,1}$, в рамках которого одновременно реализуется и максимальный для $s_{1,1}$ объем $C_{1,1}$ процессорного времени и максимальная продолжительность блокирования задачи $s_{1,1}$ низкоприоритетной задачей. В общем случае возможность такого совпадения не гарантируется, сумма $C_{1,1} + B^*(s_{1,1})$ является лишь оценкой сверху $R^*{}_{\max}(s_{1,1})$ для значения времени отклика.

Аналогично оценка снизу $R^*{}_{\min}(s_{1,1})$ для времени отклика заданий типа $s_{1,1}$ составляет

$$R_{\min}(s_{1,1}) \ge R^*{}_{\min}(s_{1,1}) = C_{\min}(s_{1,1}), \qquad (12)$$

что соответствует сценарию, при котором исполнение задания типа $s_{1,1}$ не блокируется низкоприоритетными задачами, и само $s_{1,1}$ исполь-

зует минимальный объем процессорного времени $C_{\min}(s_{1,1})$. Учитывая выражения (10), (11) в (12), получаем оценку $S^*(m_{1,1})$ для фактора сдвига $S(m_{1,1})$ первого сообщения цепочки Ψ_1:

$$S(m_{1,1}) \le S^*(m_{1,1}) = C_{1,1} - C_{\min}(s_{1,1}) + B^*(s_{1,1}).$$

Оценки факторов сдвига для некорневых задач цепочки Ψ_1 и для порождаемых ими сообщений вычисляются последовательно:

$$S^*(s_{1,k}) = S^*(m_{1,k-1}) + L_{\max}(m_{1,k-1}) - L_{\min}(m_{1,k-1}) + L_{\text{elem}}, \qquad (13)$$

$$S^*(m_{1,k}) = S^*(s_{1,k-1}) + C_{1,k-1} - C_{\min}(s_{1,k-1}) + B^*(s_{1,1}). \qquad (14)$$

4.3. Оценка времени отклика низкоприоритетных цепочек задач. Для каждой низкоприоритетной цепочки задач Ψ_j ($j \le 2$) время отклика определяется по формуле, аналогичной формуле (3), однако, при $j \ge 2$ в выражениях для $R(s_{j,i})$ и для $R(m_{j,i})$ появляются дополнительные слагаемые, отражающие фактор приоритета. Так, вместо выражения (4) при $j \ge 2$ время отклика $R(s_{j,i})$ задач представляется суммой

$$R(s_{j,i}) = C(s_{j,i}) + B(s_{j,i}) + I_{j,i}. \qquad (15)$$

Здесь фактор приоритета $I_{j,i}$ отражает максимально возможную суммарную продолжительность интервалов времени, в течение которых задача $s_{j,i} \in P$ ожидает момента предоставления ей процессора P, занятого исполнением более приоритетных задач. Так как критический сценарий неизвестен, остается неясным, существует ли такой сценарий, при котором каждая из вытесняющих $s_{j,i}$ задач τ_x активизируется в рамках интервала существования задания типа $s_{j,i}$ максимально возможное для τ_x число раз. Поэтому в рамках общей вычислительной модели возможно определить лишь оценку сверху $I^*_{j,i}$ для величины $I_{j,i}$:

$$I_{j,i} \le I^*_{j,i} = \sum C_x \left\lceil \frac{R^*(s_{j,i}) + S^*(\tau_x)}{T(\tau_x)} \right\rceil, \qquad (16)$$

где суммирование ведется по всем более приоритетным, чем $s_{j,i}$, задачам $\tau_x \in P$. Период $T(\tau_x)$ каждой из этих задач τ_x равен периоду всей цепочки Ψ, в состав которой входит задача τ_x, но каждая из таких τ_x отличается своим значением сдвига $S(\tau_x)$. Как и для Ψ_1, сдвиг корневой задачи каждой цепочки Ψ_j равен нулю при любом j.

В случае низкоприоритетных цепочек выражение (6) для оценки $R^*(m_{j,i})$ продолжительности интервалов передачи сообщений дополняется третьим слагаемым, отражающим возможность ожидания низкоприоритетной цепочкой Ψ_j предоставления услуг коммуникационной подсистемы, занятой передачей сообщений более приоритетными цепочками: $R^*(m_{j,i}) = L_{\max}(m_{j,i}) + B_{\max}(m_{j,i}) + I^*(m_{j,i})$. Фактор веса сообщения $L_{\max}(m_{j,i})$ и фактор блокирования $B_{\max}(m_{j,i})$ определяется так же, как и для сообщений цепочки Ψ_1. Оценка $I^*(m_{j,i})$ фактора приоритета определяется выражением

$$I^*(m_{j,i}) = \sum L_{\max}(m_x) \left\lceil \frac{R^*(m_{j,i}) + S^*(m_x)}{T_x} \right\rceil, \qquad (17)$$

где суммирование ведется по всем более приоритетным сообщениям, которые могут быть порождены в рамках интервала времени доставки $m_{j,i}$.

Как видно из формул (15) и (16), вычисление оценки времени отклика для задач цепочки Ψ_j требует предварительной оценки фактора сдвига для задач из более приоритетных цепочек. Аналогично, оценка времени отклика сообщений из цепочки Ψ_j по формуле (17) требует предварительной оценки $S^*(m_{x,y})$ фактора сдвига для сообщений из более приоритетных цепочек. Значения факторов сдвига для элементов высокоприоритетной цепочки Ψ_1 получаются по формулам (11) и (12), что дает возможность оценить по формулам (15–17) время отклика элементов цепочки Ψ_2. Чтобы оценить время отклика элементов цепочек Ψ_3, Ψ_4 и т.д., необходимо использовать формулы для факторов сдвига элементов соответствующих низкоприоритетных цепочек.

4.4. Оценки фактора сдвига для низкоприоритетных цепочек. При вычислении оценок значений фактора сдвига для задач и сообще-

ний из высокоприоритетной цепочки Ψ_1 по формулам (13, 14) учитываются оценки максимальной вариации фактора веса задач $C_{1,k} - C_{\min}(s_{1,k})$ и сообщений $L_{\max}(m_{1,k}) - L_{\min}(m_{1,k})$, а также оценки фактора блокирования при исполнении задач $B^*(s_{1,k})$ и передаче сообщений $B^*(m_{1,k})$. При вычислении оценок фактора сдвига $S^*(m_{j,k})$ для сообщений из цепочки Ψ_j при $j \geq 2$ необходимо дополнительно учесть вклад фактора приоритета $I^*(s_{j,i})$ в вариацию продолжительности интервала исполнения $s_{j,k}$:

$$S^*(m_{j,k}) = S^*(s_{j,k}) + C_{j,k} - C_{\min}(s_{j,k}) + B^*(s_{j,k}) + I^*(s_{j,k}). \quad (18)$$

Как видно из (16), для вычисления последнего слагаемого $I^*(s_{j,i})$ в формуле (18) необходимо учитывать значения фактора сдвига $S^*(s_{j,k})$ для задач из более приоритетных (относительно Ψ_j) цепочек. То есть, значения $S^*(s_{j,k})$ для более приоритетных цепочек должны быть вычислены предварительно. Соответствующие вычисления выполняются по формуле

$$S^*(s_{j,k}) = S^*(m_{j,k-1}) + L_{\max}(m_{1,k-1}) - L_{\min}(m_{1,k-1}) + L_{\text{elem}} + I^*(m_{j,k-1}). \quad (19)$$

Вычисления производных параметров модели по формулам (13, 14) и (18, 19) следует выполнять последовательно — сначала для Ψ_2, затем для Ψ_3 и так далее. Тогда к моменту вычислений значений сдвигов задач и сообщений цепочки Ψ_j все необходимые для этих вычислений параметры будут уже определены.

5. Заключение. Одним из свойств распределенных СРВ является наличие таких внешних событий, которые обрабатываются задачами, размещаемыми в различных узлах сети. Такие внешние события инициируют исполнение цепочек задач, обменивающихся сообщениями, передаваемыми по коммуникационным линиям, связывающими узлы в систему. Исполнение цепочки задач представляет собой чередование фаз исполнения задач с фазами передачи сообщений. Продолжительность исполнения отдельной задачи, входящей в цепочку (время отклика задачи) может определяться методами, используемыми для однопроцессорных систем с учетом факторов веса, приоритета, блокирования, а также тех или иных дополнительных факторов — например, фактора дребезга. Оценка продолжительности передачи отдельного

сообщения (оценка времени доставки сообщения) определяется методами, аналогичными методам оценки времени отклика задач. Продолжительность исполнения цепочки задач определяется как сумма продолжительностей исполнения самих задач и продолжительностей доставки передаваемых между ними сообщений. Вариации продолжительности фаз исполнения отдельных задач и передачи отдельных сообщений приводят к появлению фактора сдвига — возможной вариации смещения (относительно внешнего события) моментов активизации некорневых задач цепочки и моментов поступления в сеть передаваемых в цепочке сообщений. Появляется необходимость учета фактора сдвига при расчете времени отклика составляющих цепочку задач и времени доставки соответствующих сообщений. В условиях, когда критический сценарий системных событий неизвестен, вместо точного параметра, характеризующего максимальную продолжительность времени исполнения цепочки (времени отклика цепочки), приходится довольствоваться оценкой времени отклика цепочки. Оценка должна гарантированно не занижать истинного значения параметра. Наличие фактора сдвига приводит к тому, что оценка времени отклика значительно превышает его действительное значение, что может привести к пессимистической оценке выполнимости отдельных цепочек и приложения в целом.

Литература

1. *Liu C.L., and Layland J.W.* Scheduling Algorithms for Multiprogramming in Hard Real-Time Environment // Journal of the ACM. 1973. V. 20, no 1. P.46–61.
2. *Liu J.W.S.* Real-Time Systems. NJ: Prentice Hall, 2000. 590 p.
3. *Никифоров В.В.* Разработка программных средств для встроенных систем. С-Пб.: С-ПбГЭТУ, 2000. 103 с.
4. *Данилов М.В.* Методы планирования выполнения задач в системах реального времени // Программные продукты и системы. 2001, №4. С. 28–35.
5. *Грюнталь А.И.* Планирование заданий с синхронным стартом // Программные продукты и системы. 2010, №4. С. 19–23.
6. *Грюнталь А.И.* Локализация ресурсов вычислительных систем реального времени // Информационные технологии и вычислительные системы. 2011, №2. С. 23–40.
7. *Грюнталь А.И.* Планирование систем с асинхронным стартом // Информационные технологии и вычислительные системы. 2012, №1. С. 32–51.
8. *Гуцалов Н.В., Никифоров В.В.* Анализ выполнимости составных задач в системах реального времени // Труды СПИИРАН. 2005. Вып. 2(2). С. 437–452.
9. *Никифоров В.В.*, *Шкиртиль В.И.* Проверка корректности логической структуры многозадачных программных приложений // Известия КБНЦ РАН. 2011, №1. С. 213–218.
10. *Никифоров В.В.* Выполнимость приложений реального времени на многоядерных процессорах // Труды СПИИРАН. 2009. Вып 8. С. 255–284.

11. *Никифоров В.В.*, *Павлов В.А.* Структурные модели для анализа многозадачных программных систем // Адаптивные и интеллектуальные роботы. 2011, №1–2. С. 19–29.
12. *Никифоров В.В.*, *Шкиртиль В.И.* Маршрутные сети — графический формализм представления структуры программных приложений реального времени // Труды СПИИРАН. 2010. Вып 14. С. 7–28.
13. *Никифоров В.В.*, *Шкиртиль В.И.* Своевременность доставки пакетов сообщений в распределенных системах реального времени // Информационно-измерительные и управляющие системы. 2010, №11. С. 58–65.
14. *Никифоров В.В.*, *Шкиртиль В.И.* Оценка времени доставки сообщений в распределенных системах реального времени // Известия ВУЗов. Приборостроение. 2010, №7. С. 33–39.

Никифоров Виктор Викентьевич — д.тех.н., проф.; ведущий научный сотрудник лаборатории технологий и систем программирования СПИИРАН. Область научных интересов: программирование систем реального времени, встроенных систем, операционные системы. Число научных публикаций — 105. nik@iias.spb.su; СПИИРАН, 14-я линия В.О., д. 39, г. Санкт-Петербург, 199178, РФ; р.т. +7(848)328-0887.

Nikiforov Victor Vikentievich — Dr.Sci. (Tech.), Prof.; Senior researcher, Laboratory for Software Technology and Systems, SPIIRAS. Research interests: methods for real-time software development, embedded systems, operating systems. The number of publications —105. nik@iias.spb.su; SPIIRAS, 39, 14-th Line V.O., St. Petersburg, 199178, Russia; office phone +7(848)328-0887.

Шкиртиль Вячеслав Иванович — к.тех.н., доц.; заведующий лабораторией технологий и систем программирования СПИИРАН. Область научных интересов: Программное обеспечение систем реального времени. Число научных публикаций — 65. jvatlas@mail.rcom.ru; СПИИРАН, 14-я линия В.О., д. 39, г. Санкт-Петербург, 199178, РФ; р.т. +7(848)328-0887.

Shkirtil Vyacheslav Ivanovich — PhD (Tech.), associate professor; Dr.Sci., Head of the Laboratory for Software Technology and Systems, SPIIRAS. Research interests: software development for real-time systems. The number of publications — 65. jvatlas@mail.rcom.ru; SPIIRAS, 39, 14-th Line V.O., St. Petersburg, 199178, Russia; office phone +7(848)328-0887.

Рекомендовано СПИИРАН, лабораторией информационно-вычислительных систем, заведующий лабораторией Воробьев В.И., д-р техн. наук, проф.
Статья поступила в редакцию 21.09.2012

РЕФЕРАТ

Никифоров В.В., Шкиртиль В.И. **Оценка времени отклика цепочек задач в распределенных системах реального времени.**

Ключевое свойство программных приложений для систем реального времени (СРВ) состоит в том, что они работают «в структуре времени, определяемой ходом внешних процессов». Такое соответствие между ходом исполнения программных компонент СРВ и ходом внешних процессов обусловлено наличием ограничений на временные рамки информационных обменов с внешними процессами. Анализ программных приложений СРВ на предмет соблюдения подобных ограничений (анализ выполнимости) строится на базе соответствующих вычислительных моделей. Модели и методы анализа выполнимости программных приложений распределенных СРВ требует учета специфики таких систем. Отличительным свойством распределенных СРВ является наличие внешних событий, обрабатываемых различными узлами сети. Внешние события инициируют исполнение цепочек задач, обменивающихся сообщениями, пересылаемыми между узлами системы. Каждая задача цепочки (кроме конечной задачи) генерирует сообщение, передаваемое по коммуникационному каналу следующей задаче цепочки.

Исполнение цепочки задач представляет собой чередование фаз исполнения задач с фазами передачи сообщений. Продолжительность исполнения отдельной задачи, входящей в цепочку (время отклика задачи), может определяться методами, используемыми для однопроцессорных систем с учетом факторов веса, приоритета, блокирования, а также тех или иных дополнительных факторов — например, фактора дребезга. Оценка продолжительности передачи отдельного сообщения (оценка времени доставки сообщения) определяется методами, аналогичными методам оценки времени отклика задач. Продолжительность исполнения цепочки задач определяется как сумма продолжительностей исполнения самих задач и продолжительностей доставки передаваемых между ними сообщений. Вариации продолжительности фаз исполнения отдельных задач и передачи отдельных сообщений приводят к появлению фактора сдвига — возможной вариации смещения (относительно внешнего события) моментов активизации некорневых задач цепочки и моментов поступления в сеть передаваемых по цепочке сообщений. Появляется необходимость учета фактора сдвига при расчете времени отклика составляющих цепочку задач и времени доставки соответствующих сообщений.

В условиях, когда критический сценарий системных событий неизвестен, вместо точного параметра, характеризующего максимальную продолжительность времени исполнения цепочки (времени отклика цепочки), приходится довольствоваться оценкой времени отклика цепочки. Оценка должна гарантированно не занижать истинного значения параметра. Наличие фактора сдвига приводит к тому, что оценка времени отклика значительно превышает его действительное значение, что может привести к пессимистической оценке выполнимости отдельных цепочек и приложения в целом.

SUMMARY

Nikiforov V.V., Shkirtil V.I. **Response time estimation for task chains in destributed real-time systems.**

The key feature of programm applications for real-time systems consists in the fact that they function “in the time structure that is determined by evolution of external processes”. Such compliance of time structures for internal and external events in real-time system is stipulated by restrictions on the time intervals between data and signal information passing form the real-time system to external processes and vice versa. The analysis of real-time software applications on the subject of keeping such restrictions (feasibility analysis) is built on the base of corresponding exequtions models. As for destributed real-time systems, the models and methods for their feasibility analysis requier taking into account some specific features of such systems.

The main specific feature of distributed real-time system is the processing of external events by a chain of tasks, sited in different nodes of the system. Every external event provokes execution of such chain of tasks. The chain begins by its root task. The root task is activeted exactly at the moment of the external event registration. When root task completes execution, a message is sent through communication channel to the node, which contains the next task in the chain. Every nonfinal task in the chain sends its message to next task through communication channel.

In such a way two types of phases are alternated during task chain execution — task execution and message sending. The time, required for task evaluation (task response time) can be found in the base of methods for uniprocessor systems (such methods take into account weight factor, priority factor, blocking factor and other factors, such as jitter factor). Evaluation of message delivering time may be fulfilled on the basis of the same methods. The duration of task chain execution is the sum of all tasks’ execution and all messages sent in the chain.

However, the variation of task execution time and message sending time leads to appearance of shift factor — possibile variation of difference between the moment of first task activation and the moment of activation of any sequential task in the chain. Difference between the moment of the first task activation and the moment of any message sending in the chain can vary as well. In such a way the necessity appears for taking into account the shift factor for evaluating the task response time and message delivering time.

In the case of distributed execution the critical scenario of system events is unknown. Therefore, an approximate value of task chain execution time shall be calculated instead of exact value of this parameter. The approximate value shall be no less than exact value of chain response time. The presence of shift factor can lead to significant difference between estimation of chain response time and its real value, this difference grows with increasing of the number of tasks in the chain. This fact can causes violently pessimistic estimation of the task chain feasibility and the feasibility of the whole distributed real-time application.