エンドユーザ向け情報家電制御プログラミング環境の設計

3K－01

笹本宏†　早川栄一‡　高橋延匡‡
拓殖大学大学院　工学研究科電子情報工学専攻†
拓殖大学　工学部情報工学科‡

1．はじめに

現在、情報家電の制御のためのプログラミングインタフェースとして携帯電話に注目されている。しかし、一般に携帯電話は入出力に制約があり、プログラミングに適しているとは言えない。そこで携帯電話に適したプログラミングインタフェース、支援機能が必要になる。

プログラミング環境の利用事例としては、センサ監視による情報家電の制御、時間に沿った制御、複数家電の一括制御などがある。また特徴として、障害時に対策を行ったり、現在のインタフェースに比べて負担が軽減されるなどがあげられる。

本報告ではエンドユーザを対象とした携帯電話を用いた家電制御のためのプログラミング言語と、その支援環境、実行環境に関する設計について述べる。

2．設計方針

(1)家電制御を安全に実行する

家電制御プログラムの実行はインタプリタ上で行われるので、安全に行うことができる。

(2)プログラミングを容易にする

携帯電話でのタイプ数を押さえ、開発を容易にするために対話型のプログラミングシステムを採用する。

(3)プログラミングの自由度を確保する

細かい機能を実現するライブラリを複数用意し、それらを組み合わせて制御プログラムを開発する。

(4)携帯電話の処理の負荷を下げる

転送された制御プログラムを実行したり、システム全体の管理処理を行うサーバを用意する。

(5)動作テストを可能にする

制御プログラムを安全で効率よくテストできるように情報家電シミュレータを用意する。

3．設計

3．1．システム全体の構成

図1にシステムの全体構成を示す。

A design about programming Environment for the control of consumer electronics
Hiroshi Sasamoto, Eiichi Hayakawa and Nobumasa Takahashi
Takushoku University

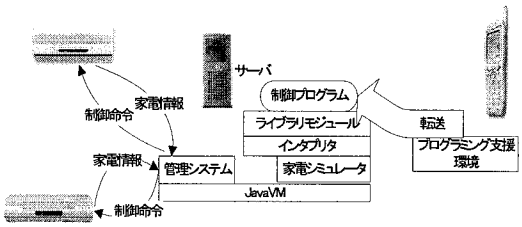

図1．システム全体構成

3．2．情報家電制御言語の設計

構文要素は

「条件」＋「情報家電名」＋「機能」＋「パラメータ」

とする。これを実行順序ごとに並べて、プログラミングする。手順ごとに並べることで、プログラミングスキルの低いユーザでもプログラミングをすることが可能になる。

条件としては、時刻、経過時間、各家電のイベント（例えば、センサの反応、動作状態のチェックなど）を用意する。条件分岐も用意する。条件分岐では分岐後は異なるタイムマップに移行し、そちらの処理が終了した時刻に本来のタイムマップに戻る。またループ機能はサポートしない。

各家電の動作は、ライブラリという形でまとめる。各家電ごとにメソッドを定義してその家電の実行を制御する。各家電ライブラリは、制御プログラムとともに、実行可能なメソッドの一覧および、パラメータのタイプ、さらに、パラメータが持つ制約情報を外部に提供する。

本言語の特徴として、各構文要素を列とする表形式で表すことが挙げられる。図2に情報家電制御言語のプログラム例を示す。

2002/03/13

条件	情報家電名	機能
10:00	エアコン	起動
	エアコン	保温運転
12:00	タッチセンサ	監視
条件	エアコン	終了

図2．制御プログラム例

表形式の構成は時間を縦軸にしており、時間を追って家電制御の様子を確認することができる。なお当然携帯電話の画面には縦方向に入りきらないが、これはスクロールして見ることになる。

次にプログラム例の説明を行う。これは 2002 年 3 月 13 日の 10 時にエアコンを起動し、「保温運転」させ、12 時にタッチセンサが監視を行い、条件が満たされたら、エアコンを終了させるというプログラムである。エアコンの機能である「保温運転」のパラメータは次の図 3 のように表される。

図 3.「保温運転」機能のパラメータ表示

「保温運転」は「暖房」モードで「20」℃を保つように運転をする設定になっている。他の機能についてはエアコンの「起動」、「終了」はエアコンの電源制御を行う。これらにパラメータは存在しない。「条件」は条件判断機能で、この場合センサの入力がパラメータとマッチしたら、その行のプログラムが実行される（つまりエアコンが終了する）。「条件」のパラメータを次の図 4 に示す。

図 4.「条件」機能のパラメータ表示

また例外処理機能も提供し、ビデオのテープがなくなった時などの例外時に行う処理を決めることができる。その処理もライブラリになっており、選択して使用する。

3．3．実行環境の設計

携帯電話で開発したプログラムはサーバに転送する。それをサーバはインタプリタ上で実行する。情報家電の機能はネットワークを介して呼び出し、シミュレータを利用する場合はインタプリタ内部から呼び出す。インタプリタ、シミュレータは Java で実現されている。またサーバは現在利用できる情報家電をネットワーク上で探し、各家電が利用できる機能ライブラリを確認し、それらを常に管理する。この機能は Jini[1]のルックアップサービスなどにも見られる。またこれらの結果を携帯電話からの要求で提供する。

3．4．プログラミングの流れ

対話型プログラミングシステムの流れを示す。

(ⅰ)サーバにアクセスして利用できる家電、機能を調べる。
(ⅱ)制御を開始する時刻を入力する。
(ⅲ)ターゲットとなる情報家電を選択する。
(ⅳ)シミュレータを利用するか選択する。
(ⅴ)その家電が利用できる機能を選択する。
(ⅵ)パラメータを入力する。
(ⅶ)例外処理を行うか選択する。
ここまでで作成完了となる。

3．5．プログラム開発環境の設計

携帯電話などのように制約されたインタフェースを持つデバイス上でプログラミングを行うには、プログラミング環境側でのサポートが不可欠である。本システムでは、次のサポートを行う。

(1) サーバと協調動作による入力支援
(2) タイムマップエディタ
(3) ライブラリのテンプレート化

(1)については、操作対象の家電が持つ属性をサーバから得て、入力を自動的に補完する機能を用意する。また、この機能では操作対象の制約（例えば、エアコンなら温度を 22℃から 30℃までの間だけに設定することや、録画などの持っていない操作を指定できないようにするなど）を与える。これによって、ユーザの間違った入力による異常動作を防ぐ。

また、時刻を軸として、各動作を編集可能にするタイムマップ機能を提供する。指定時刻における動作は、家電の操作としてもっとも多いことから、時刻ごとの動作をプログラムできるエディタを用意することで、プログラミングの手間を減らすことが可能になる。

各家電ごとの機能は、ライブラリとしてサーバに保存されている。このライブラリの開発を容易にするためテンプレートを用意する。特に、一つのメソッドに対して実際のデバイス操作とシミュレータの操作という二つの操作をペアで提供することによって、シミュレータによる動作検証を容易に行うことができる。

3．6．情報家電シミュレータの設計

シミュレータは実行環境層にインタプリタとあわせて実装する。機能の再現は行わないが、基本的に実機と同じインタフェースを提供することにより、各ライブラリによって機能を呼び出すことができる。故障などの障害を検知して、サーバに報告する機能を持っている。

4．おわりに

本報告では情報家電のプログラミングによる制御環境に関する設計を行った。今後は試作を行い、設計を煮詰めていく。

参考文献

[1] Sun Microsystems：Jini アーキテクチャの仕様
http://www.sun.com/jini/specs/japanese/jini1_1.pdf