未来に向かって

IoT 時代の環境適応型ソフトウェア

10

中川博之（大阪大学） 鄭 顕志（国立情報学研究所） 田原康之（電気通信大学）

IoT 時代のソフトウェア

近年のハードウェアの進歩に伴い，チップ，センサ，および制御装置などがますます低価格かつ高機能になってきている．その結果，あらゆる物にこれらのデバイスを埋め込み，インターネットに接続することで，データ交換や遠隔監視・制御などを実現する IoT（Internet of Things）という発想が生まれた．たとえばスマートフォンに搭載されたカメラ，マイク，GPS，加速度センサなどから得られるデータは，即座に送信することが可能である．また，センサや自動制御装置が搭載された家電製品により，いわゆるスマートハウスが実現されている．自動車の IoT 化は交通制御を飛躍的に改善する可能性があり，自動運転技術とも合わせると，これまで考えも及ばなかったような交通サービスの実現も期待できる．生産現場においても製造機器などの IoT 化により，産業全体の高度化を図るインダストリー 4.0 といった動向も注目されている．

IoT を取り巻く技術的環境は，物理世界とソフトウェアの高度な相互作用により支えられる，いわゆるサイバー・フィジカル・システム（Cyber-Physical System，CPS）であるが，IoT 時代においては，ソフトウェアがますます重要な役割を果たすこととなる．IoT においては頻繁かつ複雑な環境の変化に対応する必要があり，このような環境変化への対応を人手で行うことは困難である．したがって，人が介在することなく，動的に環境変化に適応できるような制御ソフトウェアが求められることとなる．このような背景から，システム自らが実行時に振舞いや構成を変更して，環境や実行状況の変化に対応する自己適応化技術が注目されている．

自己適応ソフトウェア

では，自己適応化とはどのように実現すればよいのであろうか．自己適応機能を有するソフトウェアは**自己適応ソフトウェア**，あるいは**自己適応システム**（self-adaptive software, self-adaptive systems）と呼ばれ，ソフトウェア工学分野において近年精力的に研究が進められている．自己適応ソフトウェアに対してはさまざまな定義がなされているが，ここでは，Ghezzi の研究グループにおける定義[1]を紹介する．この定義は，1990 年代に Zave と Jackson により示された，構築すべきソフトウェアと要求の関係式[2]に基づく．この関係式は，ソフトウェアに対する要求を R，ソフトウェアを取り巻く環境を W としたときに，構築すべきソフトウェア S は，$W, S \models R$ の関係を満たすべきであるというものである．ここで，‘$\models$’は，左辺を前提としたときに右辺が成り立つことを表す演算子である．ソフトウェアは，環境 W に対して，要求が達成された状態 R を成り立たせるような状態遷移を与える振舞い S を持つべきであることを意味している．Ghezzi らは，この関係式を発展させ，環境状態 W が W' へと変化したときにも，それを検知し，今までどおり要求 R が満たされるように振舞いを S から S' へと変更できる自律的な能力を備えたソフトウェアを自己適応ソフトウェアと定義している（**図-1**）．ここで，

- 今までの環境における関係：$W, S \models R$
- 環境変化後に望まれる関係：$W', S' \models R$

である．このような自己適応ソフトウェアを実現するためには，まず，環境の状態や自身の性能を監視（Monitor）し，監視結果をもとに現状を分析

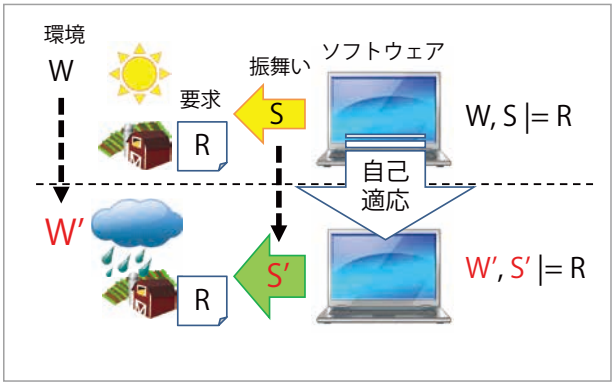

図 -1　自己適応ソフトウェア

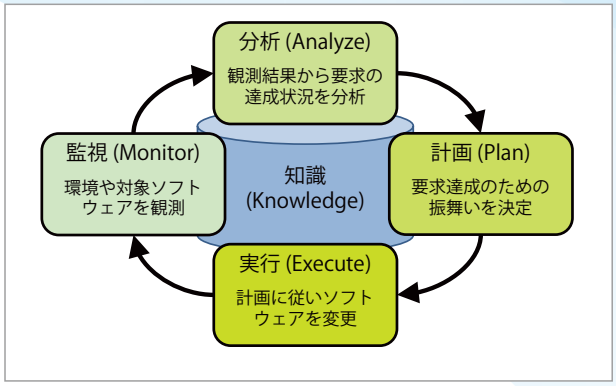

図 -2　MAPE-K ループ

(Analyze) する必要がある．また，分析結果に応じて環境変化に対応可能である一連の振舞いを計画 (Plan) し，その計画に応じて振舞いを切り替え，実行 (Execute) することも必要である．この一連のプロセスは，制御理論やロボット工学分野におけるコントローラによる実現が直感的であり，自己適応システムにおいては特に，共有知識の K (Knowledge) も組み込んで，**MAPE-K ループ**と呼ばれている (**図 -2**)．監視，分析，計画，実行の 4 つのアクティビティを繰り返すことで，動作環境の変化を検知し，その変化に適応するために自らの振舞いを変更するのである．

最近の研究動向

このような自己適応ソフトウェアに関する最近の研究動向を紹介する．

不確かさへの挑戦：IoT 時代のソフトウェアシステムは物理世界により密に組み込まれるため，デバイスの故障・誤動作，センサの観測ノイズ，観測・制御対象の変化など，システムは多種多様な環境変化に晒される．これらの変化は不確かであり，その発生を事前に予測することは難しい．そこで，不確かな環境下での保証を現実的な開発コストで行うために，従来開発時にのみ用いられていたソフトウェア工学技術を実行時にも活用する研究が進められている．

システム実行中に得られた情報をモデルに反映し，実際に起こった環境変化に対して検証・意思決定を実行時に行うことで，開発時の想定のみに頼ることのない，保証を伴った適応を実現する試みである．このために，従来は開発時に用いられていたモデル検査・確率モデル検査技術などの設計・検証技術を実行時にシステム自身が利用するよう拡張している．ただし，実行時検証は実行時のオーバヘッドが問題となるため，検証内容の可能な範囲での事前計算 [3] や投機的実行による効率化，モデル抽象化など，検証時間を抑える工夫が必要となる．

AI 系技術の応用：実行時のシステム自身による意思決定は，AI 系技術とも大いに関連する．たとえば，環境モデルの学習には機械学習が活用されている．自己適応ソフトウェアでは，実行中に得られたデータをもとに環境に起こった変化を検知し，その変化をモデルに反映しなければならない．機械学習を応用することで，実行時に得られたデータから環境モデルを構築・更新するのである．

適切な適応を決定するために，プランニングやゲーム理論も活用されている．前者は，更新された環境モデルを分析し，現在の状態から要求を満たす状態までの動作手順を自動生成するために用いられている．後者は，ソフトウェアを自プレイヤ，環境を敵プレイヤとして，敵プレイヤの選択にかかわらず (つまりは環境側でどんな不都合なことが起きても) 自プレイヤが勝利可能な，つまり要求充足が保証された支配戦略を発見するというものである．

制御理論の応用：不確かさの存在を前提とした上

で自己適応ソフトウェアによる品質維持を保証する枠組みとして，制御理論の応用も試みられている．制御理論を応用した自己適応ソフトウェア開発手法は文献 4）によくまとめられている．ソフトウェアシステムを制御対象となるプラント，維持したい品質要求を目標値，自己適応エンジンをコントローラと見立てることで，自己適応エンジンが期待する性質を満たすことを制御理論に基づいて数学的に保証することができる．

応用分野と今後の展望

すでに自己適応技術が普及している例として，クラウドコンピューティングにおけるオートスケール機能が挙げられる．計算負荷に応じて実行を担当するサーバ機器の台数を増減させるものであり，サービス提供を停止することなく，計算負荷の増減に対し適切な対応を可能としている．

そのほか，自己適応技術を活用した高度な環境適応型ソフトウェアは，次のような幅広い応用分野で有望である．まず自己適応技術の研究コミュニティでは，クラウドコンピューティングや食品分野など，本技術の活用が期待される応用例の検討に近年注力している[5]．また，セキュリティ分野も応用が期待されている分野である．現在は人手で脆弱性を発見してセキュリティパッチを開発しているが，これらの手段の完全自動化を競い合う大会である Cyber Grand Challenge[6] が，昨年米国国防高等研究計画局（DARPA）によって開催されている（**図 -3**）．さらに，近年注目されている自動運転技術分野においても，環境変化に自律的に適応する能力は必要である．

以上，本稿で紹介した自己適応化技術は，今までの人手による作業をソフトウェアにより自動化する試みである．これは，今まで明確に分けられていたソフトウェア開発プロセスと実行プロセスとの境界を取り払う挑戦であるともいえる．

図 -3　Cyber Grand Challenge 決勝の様子 [6]

参考文献

1) Calinescu, R., Ghezzi, C. et al : Self-adaptive Software Needs Quantitative Verication at Runtime, Communications of the ACM, Vol.55, No.9, pp.69-77(2012).
2) Zave, P. and Jackson, M. : Four Dark Corners of Requirements Engineering, ACM TOSEM, Vol.6, No.1, pp.1-30 (1997).
3) Filieri, A. et al. : Supporting Self-adaptation Via Quantitative Verication and Sensitivity Analysis at Run Time, IEEE TSE, Vol.42, No.1, pp.75-99 (2016).
4) Filieri, A. et al. : Software Engineering Meets Control Theory, In SEAMS '15, IEEE, pp.71-82 (2015).
5) Self-adaptive.org. Software Engineering for Self-adaptive Systems : Exemplars, https://www.hpi.uni-potsdam.de/giese/public/selfadapt/exemplars/
6) DARPA Cyber Grand Challenge, The World's First All-machine Hacking Tournament, http://archive.darpa.mil/cybergrandchallenge/

（2017 年 4 月 14 日受付）

中川博之（正会員） nakagawa@ist.osaka-u.ac.jp

大阪大学大学院情報科学研究科准教授．1997 年大阪大学基礎工学部卒業．2008 年東京大学大学院情報理工学系研究科博士課程中退．2013 年早稲田大学博士（工学）．鹿島建設（株），電気通信大学助教を経て，2014 年より現職．

鄭　顕志（正会員） tei@nii.ac.jp

1980 年生．2008 年早稲田大学大学院理工学研究科博士課程修了．2006 年同大助手に着任．2008 年より同大助教，2010 年より国立情報学研究所助教，2015 年より同准教授．現在に至る．博士（工学／早稲田大学）．

田原康之（正会員） tahara@uec.ac.jp

1991 年，（株）東芝入社．2003 年，国立情報学研究所着任．2008 年より電気通信大学准教授．博士（情報科学／早稲田大学）．エージェント技術，およびソフトウェア工学などの研究に従事．日本ソフトウェア科学会会員．