디지털 전환을 촉진하는 NoCode와 RPA

The Catalysts for Digital Transformation, No-Code and RPA

강송희
선임연구원
dellabee@spri.kr

이현승
책임연구원
hslee94@spri.kr

유호석
책임연구원
hsy@spri.kr

Executive Summary

1980년대 이후 정보화 혁명이 심화됨에 따라 이제는 SW 없이는 업무를 수행하기 힘든 디지털 전환의 시대가 되었다. 이에 따라 전문 SW개발자에 대한 수요가 폭증하게 되었고 인건비 상승과 구인난 심화와 같은 인력 관리 측면의 위험 관리 필요성이 증가함과 동시에 SW개발의 생산성을 제고할 수 있는 다양한 방법이 급속도로 발전하게 되었다. 특히 일반 현업 직원들도 최근 각광을 받고 있는 로우코드·노코드, RPA와 같은 도구를 사용하면 직접 SW를 작성하거나 작업을 자동화해 다양한 업무를 쉽게 처리할 수 있다.

이 보고서에서는 SW개발의 생산성을 향상시켜 디지털 전환을 촉진하기 위한 그간의 노력을 정리하고 로우코드·노코드와 RPA의 적용 가능성과 한계에 대해 검토한다. 생존을 위해 디지털 전환을 당면 과제로 둔 기업과 기관들은 이러한 문제 해결 보조 도구들을 검토해 볼 가치가 있다. 로우코드·노코드와 RPA는 소프트웨어적인 한계 외에도 현업 직원들이 수월하게 사용하려면 컴퓨팅적 사고력이 필요하다는 데 유의하여야 한다.

As the information revolution intensified after the 1980s, it is now an era of digital transformation where it is difficult to perform business without software. Accordingly, the demand for specialized software developers has exploded, and the risk of manpower management, such as an increase in labor costs and intensification of recruitment difficulties, has increased, and various methods for promoting the productivity of software development have rapidly developed.

In particular, by using tools such as low-code, no-code, and RPA, which have recently been in the spotlight, general business employees can easily solve problems that require software functions. This report summarizes the historical efforts to promote digital transformation by improving the productivity of software development, and examines the applicability and limitations of low-code, no-code, and RPA.

Organizations that face digital transformation to survive are well worth a look at these problem-solving aids. Low-code, no-code and RPA can be used easily if they have a computational thinking, which is a basic ability such as reading and writing, but it is necessary to accurately recognize their uses and limitations in advance.

Ⅰ 논의 배경

1. 디지털 전환의 개념과 제도적 촉진 사례

▨ 디지털 전환은 정보화를 토대로 발전한 지능정보기술*을 통해 자동화·최적화를 구현하여 가치 창출 방식을 재구성하므로 정보화의 연속선상에 있음

◦ 정보화와 정보의 생산·유통·활용을 촉진하는 점은 같지만, 디지털 전환은 지능정보기술*과 他분야 기술을 융합하여 자율적 시스템을 추구함

* 지능정보기술은 인공지능, 데이터, 사물인터넷(IoT), 클라우드 관련 기술과 이 기술들이 실제 동작하는 유무선 정보통신망 관련 기술[1]을 총칭함

◦ 디지털 전환은 기업과 국가가 보유 정보와 자산을 디지털화하고 조직구조를 혁신하며 업무 절차를 자동화함으로써 고객을 만족시키려는 노력*의 총체

* 정보 기술과 인터넷을 통해 정보를 신속하고 정확하게 처리하며 지능과 행동을 자동화하고 최적화하여 문제를 해결하고 데이터 기반 의사결정을 지원함

– (사례) 전자소송 시스템의 소송서류에 대해 ’17년까지는 종이문서를 스캔한 전자화 문서가 허용되었지만, ’18년부터는 기계가 해석, 처리, 가공할 수 있는 전자문서를 사용하도록 원칙이 변경되어 디지털 전환을 제도화함

[참고] 디지털 전환의 제도적 촉진 사례

내년부터 **전자소송에서 소장과 답변서, 준비서면 등 소송자료는 지금처럼 스캔해 제출할 수 없고 반드시 문서의 내용 검색이 가능한 텍스트 파일 형태로 제출**해야 한다. (중략) 개정 규칙은 제8조 4항에 ‘등록사용자가 작성한 전자문서(서증 제외)의 경우 부득이한 사정이 없는 한 **문자정보의 검색 및 추출이 가능한 파일**로 제출하여야 한다’는 내용을 신설했다.

현재 대부분의 전자소송 이용자들은 종이문서를 스캔해 제출하고 있다. 소송대리를 맡고 있는 변호사 사무실에서도 변호사가 소송관련 문서를 작성해 출력한 다음 직원에게 제출하라고 넘기면 직원이 이를 스캔해 전자소송시스템에 올리는 방식을 취하고 있다.

이 때문에 **판사들은 판결문 작성 때 당사자의 준비서면에 대한 주장 내용을 판결문에 그대로 옮기기 위해 많은 시간을 소비하게 되고, 키워드 검색 등도 불가능해 필요한 정보를 찾는 데 어려움을 겪어 왔다.**(이하 생략)

* 출처: 법률신문(2017.06.26.), “전자소송자료, 스캔하지 말고 ‘텍스트 파일’로”

1 지능정보화 기본법[시행 2020. 12. 10.] [법률 제17344호, 2020. 6. 9., 전부개정] 제2조 제1호 “정보”, 제2호 “정보화”, 제4호 “지능정보기술”, 제5호 “지능정보화”, 제9호 “초연결지능정보통신망”, 제10호 “초연결지능정보통신기반”을 참조

2. 전 산업 분야 디지털 전환 요구의 증대와 도전과제

디지털 전환은 산업 전반에 지능정보기술을 접목시켜 사업모델을 변화시키고 있고, 코로나19 대유행으로 이 전환이 가속화됨(KPMG, ’20)

◦ 산업간 경계가 허물어지는 빅블러 현상이 일반화되며 기업들은 새로운 경쟁 국면을 맞게 되었고 사업 환경의 불확실성이 높아지며 디지털 전환 요구가 증대됨

– 디지털 전환은 SW와 데이터 등 디지털 자산의 의미 있는 연결을 통해서 데이터 기반 의사 결정을 가능하게 하고 이는 적시성, 민첩성, 회복탄력성 등으로 표현되는 조직의 동적 역량*을 증강함

* 급격한 환경변화에 대응하기 위해 조직 내부와 외부의 역량을 통합하고 구축하여 재구성하는 능력으로 감지, 포착, 변혁의 과정을 통해 발휘됨(Teece, ’07)

불확실성에 대한 기민한 대응과 위험 관리를 위해 노동집약적 단순·반복 업무자동화와 자원 배치 최적화를 통한 기술·조직 부채 절감과 개발 생산성 제고 필요

◦ 위험 관리를 위한 기술·조직 부채 절감의 관건은 현업과 디지털 전환 전담 부서의 사업 가치 창출을 위한 원활한 협업과 조율(alignment)임

* 코로나19 대유행 이후 ‘공급망(18%)’과 ‘지역주의 회귀(14%)’를 넘어 ‘인력관리(21%)’가 조직 성장에 가장 큰 위협으로 대두했음(KPMG, ’20)

* 기술부채란 기술적으로 해결되어야 할 문제들을 뒤로 미루고 비즈니스 문제를 해결하는 시점을 앞당기는 과정에서 추후 발생할 “재작업 비용”을 의미함

◦ 멀티 스크린과 클라우드의 확산, 다양한 사업기회 탐색활동 등으로 SW개발 수요가 폭증해 SW개발 생산성 이슈를 해결할 필요도 증대됨

– 퍼블릭, 프라이빗, 하이브리드가 공존하는 멀티 클라우드 환경에 대응하면서 기존 상용SW보다 증대된 회복성, 관리 편의성, 가시성, 확장성이 요구됨

– 언제든지 다양한 정보기기에서 발생하는 각종 데이터를 합법적이고 효율적으로 수집·분석하고 보다 지능화된 서비스를 제공해야 할 필요성도 대두

3. SW개발자 부족과 SW개발 생산성 이슈

SW개발 수요 증가는 SW개발의 어려움을 환기시키고 있으며, 전 세계적인 SW개발자 부족 현상을 초래하고 있음

◦ (SW개발의 어려움) 원하는 대로 동작하는 SW를 개발하는 것이 너무 어려워서 SW개발 과정을 체계화하려는 SW공학이 등장했으나 한계가 있음

– (SW 위기) 정확하게 동작하고, 개발자가 이해할 수 있으며, 검증 가능한 SW를 개발하기가 근본적으로 어려움을 'SW위기'라 일컬음(NATO, '68)

– (SW 공학) 이에 개발과정을 체계화하여 한정된 자원으로 정해진 기간 내 고품질의 SW를 개발하려는 'SW공학'이 탄생하여 여러 방법론과 도구가 발전하게 되었음

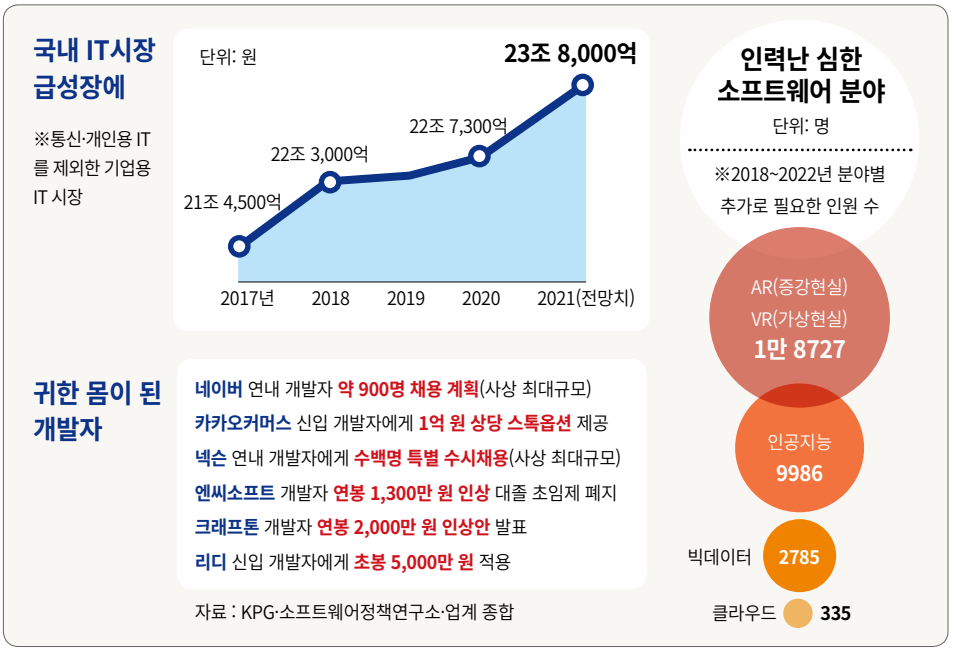
– (노력의 한계) SW개발은 창의적인 지식노동 중 하나로 각 SW개발자의 역량에 따라 성과의 차이가 뚜렷하여 아직 SW개발의 소요 비용, 기간을 정확하게 예측하기 어렵고 오류 없이 수정·보완하는 체계의 구축이 어려움

◦ (SW개발자 부족 현상) 전 세계적으로 지능정보기술 전문가를 포함하여 숙련된 SW개발자가 부족하여 SW개발자의 몸값이 급상승 중임

– 미국은 컴퓨터 관련 일자리가 '29년까지 약 53.1만개 증가하지만, 컴퓨터 공학 전공 졸업생은 연 4.7만 명으로 퇴직자를 고려하면 SW개발자 부족현상이 지속될 전망(미국 노동통계국, '19)

– 국내에서도 최근 신흥 SW기업들이 SW개발자 연봉을 급격히 인상했고(중앙선데이, '21), 대기업들은 파격적인 대우를 약속하며 인공지능 등 전문가 영입에 박차(매일경제, '21)

[그림 1] 국내 SW개발자 구인난과 연봉상승



출처 : 중앙선데이(2021.04.03.), '네카라쿠배' IT개발자 모시기 경쟁, 1억 스톡옵션도

국가 차원에서 디지털 전환을 촉진하려면 SW개발 생산성 향상을 위한 그간의 노력과 한계를 재조명해 우리가 어느 수준에 와 있는지, 전 산업을 아울러서 기업과 기관들은 어떤 도구를 쓸 수 있는지 정리하고 정보를 제공할 필요 있음

◦ 숙련된 SW개발자를 단기간에 충분히 양성하기는 어려우므로 SW개발 생산성을 향상시키기 위한 기존의 여러 성과들을 같이 활용하여야 함

◦ 아울러 디지털 전환을 달성하려면 각 기업과 기관들의 전체적인 '전환'이 필요하므로 기존의 업무를 수행하던 현업 인력이 SW개발자의 역할을 겸비할 수 있는 방안도 같이 추구하여야 함

II SW개발환경의 변화와 생산성 향상의 역사

1. 새로운 개발수요 : 웹프로그래밍, 클라우드. 멀티 플랫폼 SW

텍스트 위주의 초기 인터넷과 달리 그래픽을 지원하는 월드와이드웹이 '90년대 등장했고 동적인 웹사이트와 웹앱을 개발하는 웹 프로그래밍 개념이 정립됨

◦ 초기 웹사이트는 사람이 직접 작성한 HTML 문서와 미디어 파일들로 단순하게 구성됐으나 차츰 사용자와의 상호작용에 따라 동적으로 웹페이지를 구성하게 됨

– 동적 웹 구성을 위해 CGI, 자바스크립트 등 다양한 기술과 표준들이 등장

◦ 전통적인 프로그래밍은 운영체제에 별도로 설치하여 동작하는 SW 개발을 위한 것이었으나 웹앱과 하이브리드앱의 등장으로 웹프로그래밍이라는 분야가 정립

– 모바일 정보기기에 내장된 웹브라우저의 다양한 기능을 SW 개발에 이용하려는 시도가 웹앱*을 탄생시킴

* 웹앱은 웹 브라우저 기능을 주로 이용하는 응용SW로 모바일 기기의 어플리케이션 스토어에서 일반 응용SW처럼 내려 받아 설치 후 사용가능함

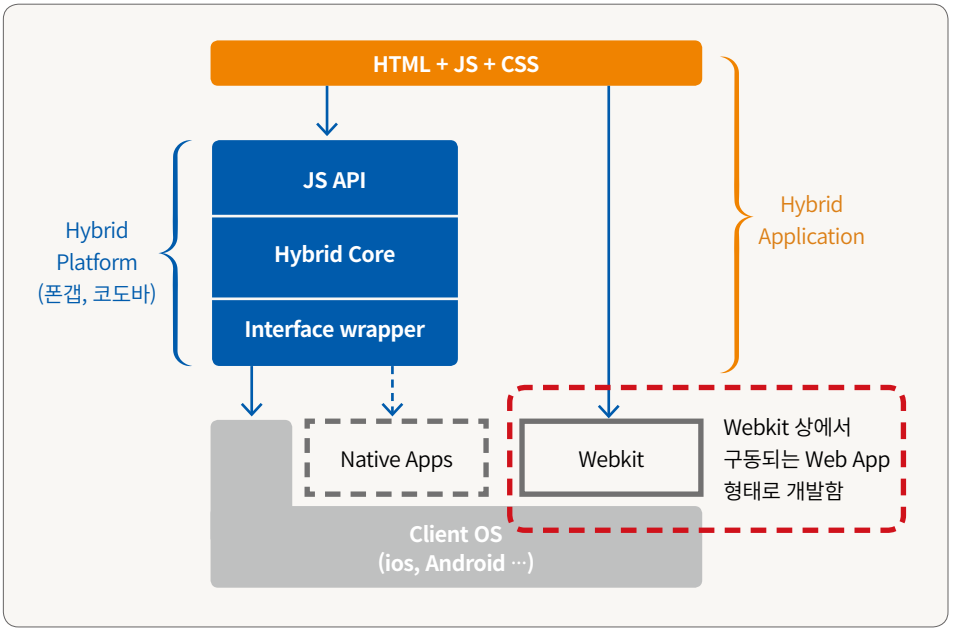
◦ 웹 브라우저와 운영체제의 기능을 같이 사용하는 하이브리드앱 개념과 지원 개발환경이 운영체제용 응용SW인 네이티브앱과 함께 보편화됨

[표 1] 모바일용 SW 개발 방법 비교

구분	모바일 웹	모바일 웹앱	하이브리드 앱	네이티브 앱
개요	• 일반 웹 기술 개발 • 모바일 브라우저에 의해 실행	• 모바일 웹의 한 형태 • 앱 형태로 패키징	• 일반 웹 기술 개발 • 네이티브 앱 활용	• 플랫폼에 따른 프로그래밍 언어와 각종 라이브러리로 개발
그래픽 지원	하	하	상	상
앱스토어판매	불가능	가능	가능	가능
매쉬업	가능	가능	가능	불가능
멀티 플랫폼	용이	용이	용이	어려움
스토리지	서버, 클라우드	서버, 클라우드	로컬, 서버, 클라우드	로컬
디바이스제어	불가능	불가능	용이	용이

출처 : 전자정부 표준프레임워크 강의

[그림 2] 하이브리드 앱의 구조



출처 : 전자정부 표준프레임워크 강의

클라우드의 보급도 새로운 SW개발 수요를 유도함

◦ 초고속 인터넷과 서버 가상화 기술을 토대로 다양한 정보자원을 빌려 쓸 수 있는 클라우드가 가격경쟁력과 편의성으로 조직 업무환경을 바꾸며 확산됨

◦ 다양한 클라우드(퍼블릭, 프라이빗, 하이브리드)가 함께 사용되는 동적인 환경에 대응 가능한 어플리케이션인 클라우드 네이티브 앱의 필요성이 증대

* 기존 상용SW보다 가용성, 확장성, 회복탄력성, 관리 편의성, 가시성, 자동화 등의 비 기능적 요건의 요구수준이 까다롭고 설계 방식이나 표준에 차이가 있음

다양한 고객의 상황에 모두 대응하려면 SW를 다양한 운영체제용으로 개발할 뿐 아니라 웹사이트로도 같은 서비스를 제공하는 멀티 플랫폼 전략이 필수임

◦ 멀티 플랫폼 전략에서 중요한 것은 표준을 준수하는 것인데 웹 표준을 준수하면 웹 사이트, MS의 윈도우, 구글의 안드로이드와 애플의 iOS 등 다양한 운영체제에서 모두 동작하는 SW를 빨리 개발할 수 있음

2. 소스코드 생성 보조 도구

SW를 개발할 때 일부 작업들을 자동화하여 생산성을 올리는 RAD[2] 개발도구, 모델링을 통해 오류 없이 신속한 개발을 추구하는 MDD[3]방법론이 탄생함

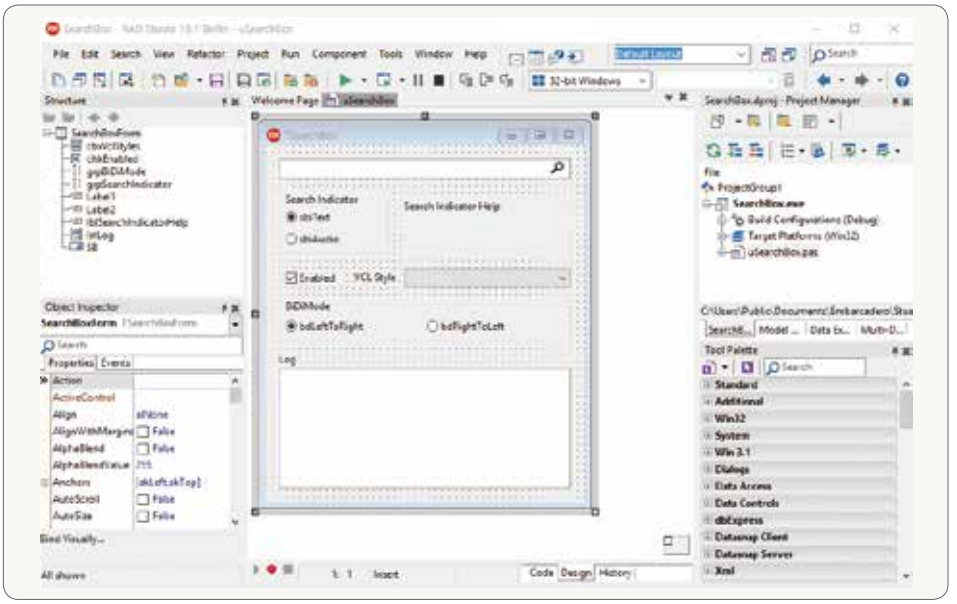
◦ (RAD 개발도구) 그래픽 사용자 환경이 보편화되자 버튼, 스크롤바, 입력창 같은 사용자 인터페이스를 빠르고 쉽게 제작하는 델파이와 같은 도구가 등장하면서 개발자가 핵심적인 알고리즘 설계와 코딩에 집중할 수 있게 됨

– 최종사용자가 SW의 외관을 빨리 확인할 수 있는 프로토타입 개발을 중시하는 RAD 방법론과 밀접한 관련을 가짐

2 “신속한 응용SW 개발”을 의미하는 Rapid Application Development의 줄임말이며, 지원하는 프로그래밍 언어가 제한되는 경우도 있음

3 Model Driven Development의 줄임말로 우리말로는 “모델 주도 개발”로 변역할 수 있음. MDD에 관한 보다 자세한 내용은 진회승, 김태호(2015), “MDD(모델 주도 개발) 유용성 논의와 사례 분석”, SPRi 이슈리포트 2015-012호 참조

[그림 3] RAD도구를 활용한 대화상자 디자인 예시



출처 : 엠바카데로사의 RAD Studio IDE

◦ (UML[4]과 MDD) SW개발 시 설계가 중요하다는 점에 착안해 모델링 언어로 작성된 설계도를 만들고, 해당 모델을 SW도구로 소스코드로 변환하여 SW를 개발하려는 시도가 등장함

– (UML) SW에서 각종 데이터를 저장하고 동작을 수행하는 객체(Object)를 식별하고 그들 간의 관계를 표현하기 위한 모델링 언어로 '90년대 중반부터 개발된 국제표준으로 점점 용도가 확장되었음

– (MDD) 금융, 제조 등 각 분야에서 목표하는 업무 프로세스를 UML 또는 도메인 특화 언어 등으로 모델링한 후 필요한 산출물과 소스코드를 자동으로 생성하는 개발방법을 말함

RAD 개발도구, UML과 MDD의 장단점

◦ (RAD 개발도구) 그래픽 사용자 환경의 SW개발이 편리해 졌으나 개발된 SW의 성능이 저하되는 단점과 SW개발자의 보조도구인 측면이 강함

– RAD로 개발한 SW는 그렇지 않은 SW에 비해 메모리 요구량이 커지거나 동작 속도가 느린 편이어서 이를 개선하기 위한 노력이 지속 추진됨

4 통합 모델링 언어(Unified Modeling Language)는 소프트웨어 공학에서 사용되는 표준화('97)된 범용 모델링 언어임

– 프로그램 완성을 위해서는 프로그래밍 언어로 소스코드를 작성해야 하는 부분이 남아 있어 기존 SW개발자의 보조도구인 측면이 강했음

◦ (UML과 MDD) SW설계가 가시화되고 산출물이 신속하게 확보되는 장점이 있었으나, SW개발자가 모델링 언어를 새로 익혀야 하며 자동 생성된 소스코드는 인간인 SW개발자가 이해하기 어려운 단점이 있었음

– UML 등으로 표현된 업무 프로세스를 현업 담당자가 같이 리뷰하기 때문에 요구사항이 명확해져 SW개발 프로젝트의 성공률을 높일 수 있음

– 모델링 언어와 MDD 도구를 수정 또는 자체 개발해야 하는 경우도 있어 모든 SW 개발에 적합하다고 보기 어려우며 기업 업무에 특화된 편임

3. 비즈니스 프로세스와 데이터베이스 관리 보조 도구

비즈니스 프로세스 관리 보조 도구 : BRE(Business Rule Engine)

◦ 기업 업무 프로세스 변경을 자동화하기 위해 BRE와 BPM, BAM을 조합

* BPM : Business Process Modeling, BAM : Business Activity Monitoring

– 부서별 업무 흐름 변경, 금융 상품 개발 등에 활용됨

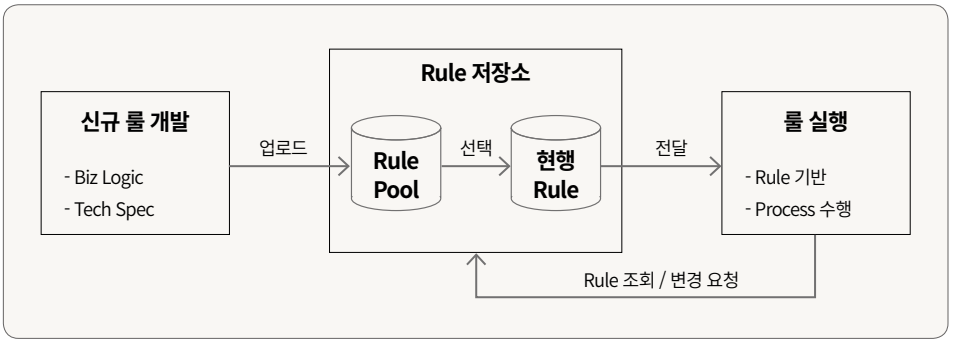
* 예시) 전결규정 등 결재단계를 BPM으로 모델링하고 BRE에 저장한 후, BAM으로 결재단계별 소요시간을 모니티링하면서 개선이 필요한 경우 BRE에서 결재단계를 변경하면 자동으로 반영됨

[표 2] BPM, BRE, BAM 의 역할 및 특성 비교

구분	BPM	BRE	BAM
개요	BPM 프로세스모델링 → (모델링) BRE 모델기반 룰정의 ⇄ (룰 수행 / 룰 조정) BAM 모니터링·관리		
목적	프로세스 통합	룰정의, 자동화	프로세스 관리
관심대상	EndToEnd 프로세스	비즈니스룰	KPI, 위험요소
기능	프로세스 최적화	비즈니스룰 실행	프로세스 모니터링
특징	통합용 SW	자동화 SW	업무 관리용 SW

◦ BRE는 비즈니스 프로세스 규정을 파일(DB)의 형태로 저장하며 규정이 변경되어 해당 파일을 수정하면 코딩 없이 바로 비즈니스 프로세스가 변경됨

[그림 4] BRE 구조



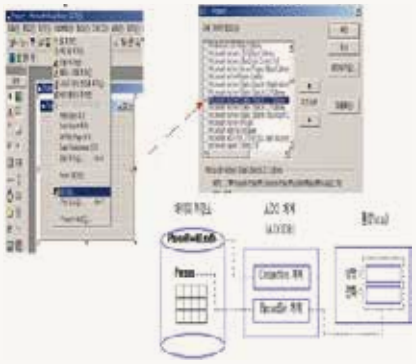
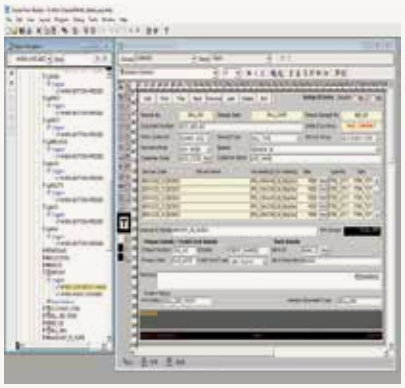
◦ 사전 업무 프로세스 컨설팅, 광범위한 시스템 통합, 높은 솔루션 도입 등에 많은 비용과 시간이 소요되는 한계가 있었음

데이터베이스 구축 보조 도구

◦ 데이터베이스 공급 기업이 DB구축을 보조하는 GUI 기반 도구를 제공하였으나, DB중심의 접근으로는 프로세스 자동화에 한계가 있음

[표 3] 4세대 RAD와 데이터베이스 구축 보조 도구 비교

구분	4세대 RAD 도구	데이터베이스 구축 보조 도구
특성	• 소스코드 생성 보조 • DB 테이블과의 연계를 제공 • 테이블 생성, 변경, 관리기능을 일부 제공하나 DB구축이 자동으로 이루어지는 수준은 아님	• 소스코드 생성 보조 • DB 테이블과의 연계뿐 아니라 데이터베이스 구축 자동화 • HTML 등 웹프로그래밍에 적합
한계	• 웹프로그래밍으로는 적합하지 못하고 노코드와 같은 수준의 자동화 수준에는 이르지 못함 • 당시의 객체지향, 컴포넌트 기반 개발(CBD, Component based Development), 서비스 지향설 계(SOA, Service-oriented Architecture)로 이 어지는 기술 패러다임의 변화에 적응하지 못했음	• DB 중심의 웹프로그래밍에 적합했으나, 데이터와 비즈니스 프로세스를 엄격히 구분하는 CBD, SOA 등의 기술 패러다임의 변화에는 적응하지 못함

구분	4세대 RAD 도구	데이터베이스 구축 보조 도구
개요		

4. 디지털 전환을 촉진하는 새로운 접근법

현업에서 직접 시각화 도구, 드래그 앤 드롭 방식으로 프로토타입 제작이 가능한 로우코드·노코드와 기존 SW들의 기능을 활용해 자동화하는 RPA 스크립트는 현업과 IT 전담 부서 간의 협업과 소통을 돕고 디지털 전환 역량 확보를 촉진

[그림 5] 자동화, 최적화 등 디지털 전환 역량 확보를 위한 도구

출처: Gartner('20), Top 10 Trends in PaaS and Platform Innovation, 2020 참조 저자 수정

◦ (로우코드·노코드) 클라우드 기반으로 출현한 시각적 모델링과 선언적 기술*로 앱을 신속하고 쉽게 개발하는 환경을 제공하는 도구(Forrester, '14)이며 신기술과 새로운 해결책의 탐색, 개념 증명 등에 사용됨

* 드래그 앤 드롭 디자인 기능과 선언·설정하는 방식의 데이터 모델링 등을 제공하는 것으로, 코드 보다 시각적 도구를 중시하게 됨

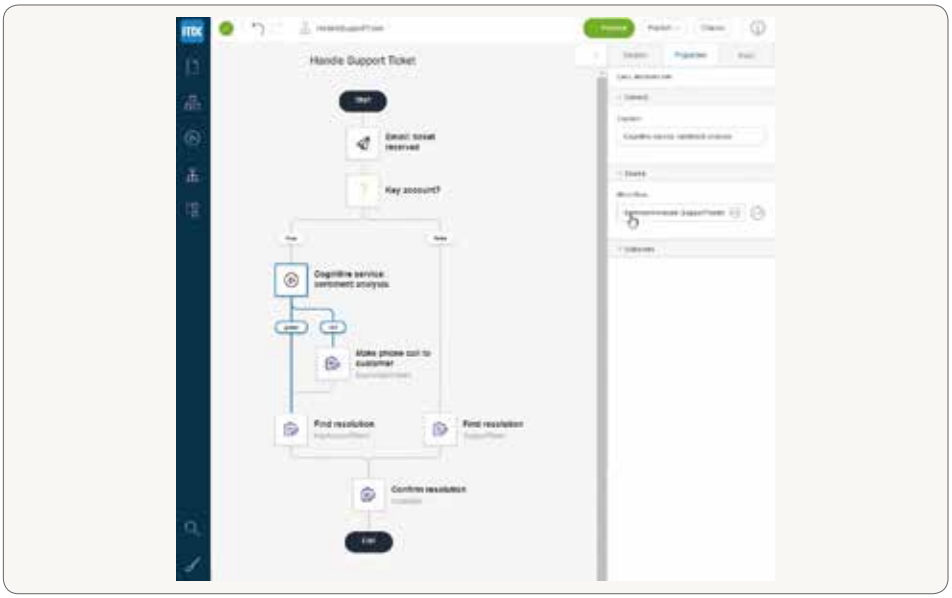
– 로우코드, 노코드 플랫폼을 로우코드 플랫폼이라 통칭하며, 특히 노코드 플랫폼은 일반 사용자도 코딩 관련 사전지식 없이 드래그 앤 드롭 등을 통해 응용SW 구성 요소를 추가하도록 하여 앱을 만들 수 있도록 함

[표 4] 로우코드와 노코드 비교

구분	로우코드	노코드
대상	일부 개발경험, 프로그래밍 지식이 있는 파워 사용자나 앱을 빨리 구축해야 하는 전문SW 개발자	프로그래밍 경험이 없는 현업직원, 일반 시민들까지 대상에 포함
기능	모델 기반, 메타 데이터 기반 프로그래밍 언어 등 선언적 기술과 고수준 프로그래밍 언어 추상화, 원스톱 배포 등을 지원	코딩이 전혀 포함되지 않는 시각적 드래그 앤드 드롭 인터페이스를 제공 * 커스터마이즈 관점에서는 로우코드보다 제한
구성 요소	시각적 개발 환경, 백엔드 시스템, 데이터베이스, 웹서비스나 API에 대한 자동화 링크 등	

출처 : Forrester('20), IDG('17), 언론 등을 종합 저자 작성

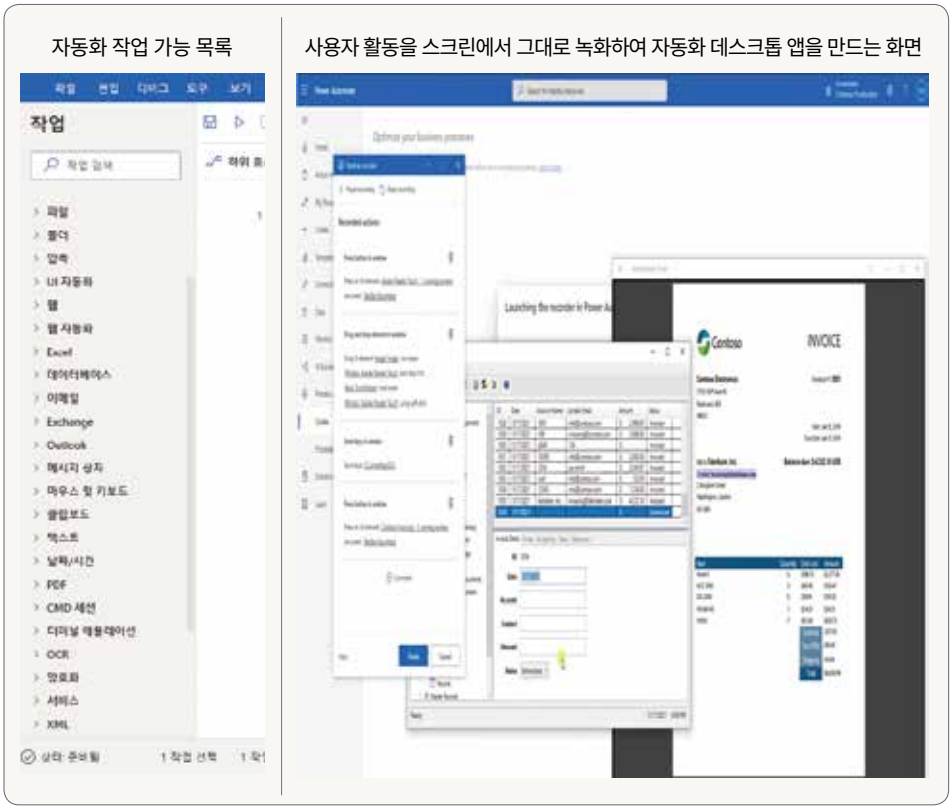
[그림 6] 로우코드 모델링 화면 예시



출처 : https://www.mendix.com/workflow/, 2021.05.27. 접속

◦ (RPA) 기존 환경에서 작동하는 다양한 SW들을 통합할 수 있는 RPA(Robotic Process Automation)는 스크립트, 매크로 등 논리 규칙을 통해 저비용·낮은 복잡도로 반복 업무를 자동화하여 민첩성과 효율성을 확보 가능

[그림 7] RPA 도구 – MS 파워오토메이트 실행화면



* RPA의 산업 전반의 도입률은 ’20년 기준 22%로 인공지능 기술 중 가장 높은 컴퓨터 비전(18%), 딥 러닝(16%)보다 높음(맥킨지, ’20)

* 규칙기반 업무, 시스템 연계, 데이터 추출 및 가공 등의 20여개 업무에 RPA도입 후 73% 효율성이 증가한 사례가 보고됨(김기봉, ’19)

◦ RPA는 보통 별도 SW가 존재하지만 상용SW에 내장되기도 하며, ‘데스크탑 자동화(RDA, Robotic Desktop Automation)’, ‘프로세스 자동화’, 인공지능 기술과 결합한 ‘지능적 자동화’ 등으로 세부적으로 구분되기도 함

* RPA는 인간의 행동과 작업을 자동화하고, 인공지능은 지능과 추론을 모방해 분석, 의사결정을 자동화하므로 서로 결합 가능함(IEEE 표준 2755-2017)

[표 5] RDA, RPA의 비교

구분	RDA(Robotic Desktop Automation)	RPA(Robotic Process Automation)
관리방식	직원에 의해 필요 시 PC에서 실행되는 개인 비서 개념으로, 수시로 필요한 자동화 업무를 직원이 직접 수행시켜 결과를 얻음	중앙 집중 관리되는 SW봇으로 상호작용하는 사용자가 없으며 주기적으로 수행되고 결과 등을 확인하는 운영자가 필요함
실행권한	직원 PC에 여러 자동화 업무 실행파일 존재	RPA매니저 등 운영·관리자에게 실행권한이 있음
장점	개인 맞춤형으로 RPA보다 낮은 비용	직원의 개입 없이도 24시간 예약된 작업을 실행

출처: Tobeware('19), RPA

로우코드·노코드와 RPA는 새로운 정보시스템 구축 또는 주문형SW개발 사업 없이도 저렴하게 업무효율화를 달성할 수 있어서 각광받고 있음

◦ 클라우드가 확산되고 SW개발자 구인난이 이슈화되면서 현업 직원이 직접 로우코드·노코드와 RPA를 활용하여 문제를 해결하려 하는 요구가 증대함

◦ 또한 주 52시간으로 대표되는 업무시간 단축과 일-가정 양립을 추구하는 흐름 속에서도 자동화를 통한 업무효율화로 인건비 증가를 최소화하기 원함

III 로우코드 · 노코드 : 신규 개발, 탐색을 위한 도구

1. 로우코드·노코드의 효과 및 이점

(출현 배경) SW개발 역량이 없는 현업직원이 신속하게 앱을 개발하여 사내에서 사용하거나 외부 고객에게 신속하게 제공할 수 있도록 하기 위해 코딩을 최소화하거나 아예 없앤 개발도구들이 출시되어 왔음

◦ 해당 업무와 조직의 요구사항을 잘 아는 현업직원이 조직 내 IT부서나 외부 SW기업에 의존할 필요 없이 앱을 개발 및 개선할 수 있음

– 로우코드 개발은 기본적인 프로그래밍 지식은 필요하며, 전문 SW개발자와 다른 기술 역량이

나 배경을 가진 사람(데이터 분석가·SW관리 책임자·비즈니스 분석가 등)이 대다수

◦ 최근에 출시된 로우코드·노코드는 클라우드(aPaaS) 기반으로 기획·구현·테스트 등 전체 개발과정을 지원하고 위험 관리 기능, 모범 사례까지 제공해 개발 편의성을 높이고 SW개발에 드는 시간, 비용을 절감함

– 초기 현업 직원들의 개발 도구(Civil Development Tools)는 광고 대비 미비한 기능, 모범·표준 사례 미지원, 취약한 보안 등으로 제한적으로 활용됨

[표 6] 기존 스크래치 개발 방식과 로우코드 개발 방식의 비교

<table>
<tr><th>구분</th><th colspan="2">기존 SW 개발</th><th colspan="2">로우코드·노코드 개발</th></tr>
<tr><td rowspan="6">개발 과정</td><td>1</td><td>요구사항 수집 및 분석</td><td>1</td><td>요구사항 수집 및 분석</td></tr>
<tr><td>2</td><td>아키텍쳐 설계</td><td>2</td><td>로우코드 플랫폼, API 선택</td></tr>
<tr><td>3</td><td>백엔드/프론트엔드 프레임워크, 라이브러리, 배치 스택 등 선택</td><td rowspan="2">반복 수행</td><td>워크플로우, 데이터 모델, UI 모델링 (시각적 IDE를 주로 이용)</td></tr>
<tr><td>4</td><td>직접 UI, 로직을 정의하고 구현·코딩</td><td>기능검증·성능테스트</td></tr>
<tr><td rowspan="2">반복 수행</td><td>기능검증·성능테스트</td><td rowspan="2" colspan="2"></td></tr>
<tr><td>업데이트 및 수정</td></tr>
<tr><td>특장점</td><td colspan="2">• 전통적인 점진적 개발 방식
• 변경이 잦지 않다면 꼼꼼한 관리, 통합 및 보안 정책 적용 가능</td><td colspan="2">• 며칠, 몇 주 만에 프로토타입을 만들고 계속 보완하면서 앱을 완성해 생산성 최대 20배 향상 (Oracle, ‘21)
• 클라우드 기반으로 업데이트, 보안, 안정성에 대한 우려를 최소화함</td></tr>
</table>

출처 : IDG(2017), 로우코드

(활용 효과) 숙련된 SW개발자도 앱을 더욱 빠르고 안정적으로 개발할 수 있고, 비숙련자인 현업직원도 아이디어를 SW 형태로 구체화할 수 있어 서비스업, 제조업 등 전통산업의 디지털 전환 가속화에 직접적으로 기여하고 있음

◦ (적시성) 현대의 조직들은 운영 효율성 개선, 혁신 촉진 및 수익을 창출하기 위해 더 많은 앱이 필요하므로 도메인 지식이 있는 현업직원이 로우코드·노코드를 통해 조직에 필요한 솔루션을 더 빠르게 적시에 구축 가능함

– 특히 다양하게 세분화된 기능이 필요한 기업용 모바일 앱 시장에서 확산되고 있음

– 지난 40년간 개발된 앱의 수와 동일한 수준인 5억 개 이상의 앱이 ’23년까지 개발될 전망 (IDC, ’19)

◦ (생산성) 현업부서는 IT부서와 요구사항 상세화를 위한 소통비용을 절감하면서 간단한 앱은 직접 해결하는 한편, IT 부서는 기존 환경의 현대화, 핵심 시스템 확장 등 복잡한 고난이도 프로젝트에 핵심 역량을 집중할 수 있음

– 기존에 하나의 SW를 개발하는데 정규직 개발자 5명이 필요하고, 9개월이 소요됐다면, 로우코드·노코드를 도입했을 때 최소 4배의 생산성 향상 효과(Mendix, ’21)

◦ (위험관리) 숙련된 개발자 부족, IT 부서와의 협업 장벽, 클라우드의 확산으로 그림자 IT*가 증가하면서 품질·보안 위험*이 급증하였으나 로우코드·노코드를 공식적인 위험 평가·통제 방안과 같이 사용하면 그림자 IT의 문제는 해결 가능

– 그림자 IT란 조직 내 비IT 부서들이 자체적인 예산으로 IT부서의 승인 없이 IT프로젝트를 추진하거나 구매하는 것으로 전체 IT예산의 40-50%에 달했으나(Gartner, ’17, Everest Group, ’17), 코로나19로 인해 59%로 비중이 급증함(Core, ’21)

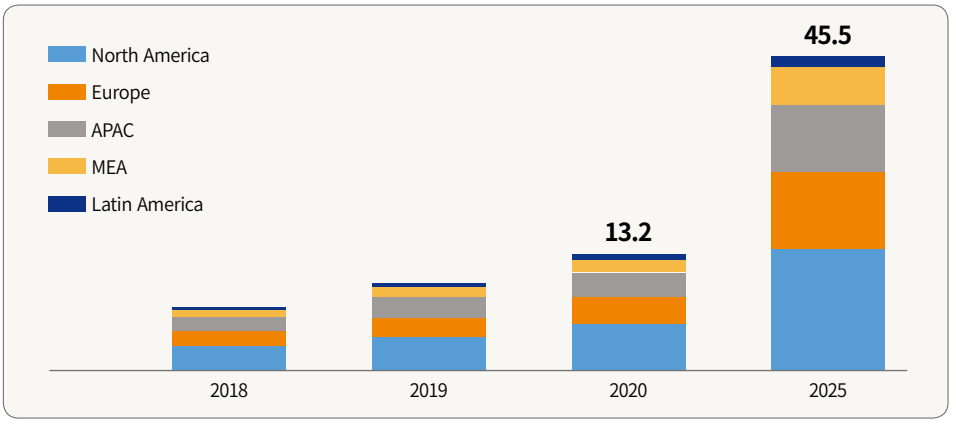
– 품질, 보안 위험으로 인한 가동 중지 시간이 1시간 이상인 경우 10만 달러 이상의 비용이 소요됨(ITIC, ’17)

2. 로우코드·노코드의 시장 지형과 전망

로우코드·노코드 시장은 ’20년 132억 달러 규모에서 ’25년 455억 달러 규모로 연평균 28.1% 성장할 것으로 전망(Markets and Markets, ’20)

◦ 로우코드 개발은 ’24년까지 모든 앱 개발 활동의 65% 이상을 차지할 것으로 예측됨(Gartner, ’20)

[그림 8] 지역별 로우코드 개발 플랫폼 시장(단위: 1B$, USD)



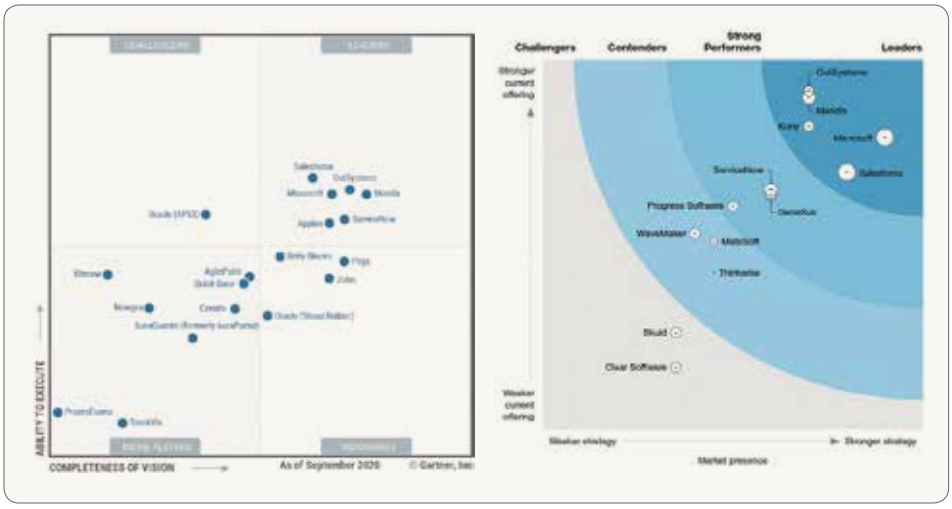
출처 : Markets and Markets(’20)

글로벌 주요 기업들은 이미 ’14년 이후 클라우드 기반 로우코드·노코드 개발도구를 출시해서 상당히 발전시켰으나 국내는 태동기에 들어섬

◦ (해외) 글로벌 기업들은 다양한 수준의 사용자들을 대상으로 로우코드·노코드 기능을 모두 제공하며 클라우드, 온프레미스 모두 제공되는 제품도 존재함

– 마이크로소프트(파워앱스), 세일즈포스(플로우), AWS(허니코드), 아웃시스템즈, 멘딕스, 오라클(APEX), 페가시스템즈, 조호 등

[그림 9] 시장 참여자 지형도(좌: Gartner(’20), 우: Forrester(’20))



◦ (국내) 국산 플랫폼으로는 로우코드를 표방하는 CALS, 노코드 플랫폼 아가도스 등이 있으며 태동기로 볼 수 있음

3. 로우코드·노코드 도입의 고려사항

SW 생명 주기 전반의 관리를 지원하며 현업과 IT 부서간의 협업과 소통을 원활하게 하면서도, 회복 탄력성, 확장성 및 보안성을 지원할 수단을 제공해야 함

◦ (앱 생명주기 지원) 앱의 개발·운영·성능 관리의 전 생명주기의 지원 가능성

◦ (사용자 경험) 몰입형, 대화형 통합 개발 환경을 웹/모바일로 제공하여 생산성과 편의성을 향상해야 함

◦ (협업 기능) 숙련된 개발자와 현업 직원을 각각 식별해야 하며 이들의 실시간 협업을 돕는 작업·메시지 로깅 및 승인, 충돌 해결이 가능해야 함

◦ (데이터 통합) 데이터 보안 위협과 손실을 방지하면서도 모든 소스에서 데이터를 쉽게 찾고 해석과 사용이 가능해야 함

◦ (클라우드 지원) 최종 산출물 앱의 회복 탄력성, 확장성 및 보안성을 유지하기 위해 클라우드 네이티브 앱 개발이 가능해야 함

[그림 10] 로우코드·노코드 기술과 제품 평가 요소

4. 로우코드·노코드의 용도와 한계

(제한된 응용분야) 특수하거나 복잡한 고난이도의 프로그래밍 기술이 필요하지 않고 표현 계층에 더 많은 노력과 예산이 소요되는 응용 분야에 적합

◦ 규칙 기반으로 모델링할 수 있는 응용 분야에 모두 적용 가능하나 주로 탐색적인 신규 개발 및 프로토타입에 활용됨

◦ 웹사이트나 모바일 앱, 조직 내부 생산성 제고를 위한 앱, 템플릿 제작, 지불·결제, 전자상거래, 회계 모델링, 예측, 데이터 분석 등의 분야에 적용

(한계) 보수적인 운영 요건과 엄격한 성능 요건이 있는 환경에서는 사용되기 어렵고, 도입 기관은 플랫폼에 종속될 가능성이 높음

◦ 노코드의 경우 최소한의 교육을 받지 않은 현업이 전문부서 검토나 승인 없이 사용하면 비즈니스 로직 상 데이터를 노출하는 등 보안 문제나 규정 준수·통합 이슈가 발생하거나 앱이 과도한 리소스를 사용하여 전사 시스템에 영향을 줌

– 조직·기술부채를 최소화하기 위해 그림자 IT를 공식화할 목적으로 노코드를 도입하더라도 전사 시스템 아키텍쳐와 보안 정책에 대한 검토와 교육·승인이 없으면 오히려 기술부채가 더 증가할 위험도 있음

◦ 로우코드·노코드 플랫폼으로 비즈니스 앱을 개발할 경우 플랫폼에 종속되는 경우가 대부분이므로, 초기 도입 시부터 이관 가능성·멀티 클라우드 환경을 고려할 필요가 있음

Ⅳ RPA : 기존 SW들의 통합

1. RPA의 효과 및 이점

(출현 배경) 더 적은 자원으로 더 많은 작업을 수행하고 싶은 기업·기관에서 단순반복적인 행위로 구성된 업무를 전용 SW를 개발하지 않고 낮은 비용으로 스크립트 파일[5]같은 형태로 자동화를 지원하기 위해 개발된 SW임

◦ 2000년대 초반에 스크린 스크래핑 기술*에 토대하여 SW봇을 쉽게 제작하는 방식을 본격적으로 소개한 업체인 블루프리즘이 RPA라는 용어를 처음 사용함

* 스크린 스크래핑(scrapping)은 컴퓨터화면에서 텍스트 데이터를 읽는 기법으로 현재는 주로 웹 환경에서 인간을 위한 출력형태의 데이터에서 다시 구조화된 데이터를 추출하여 변환하는 기술을 지칭함

5 SW 분야에서 스크립트 파일이란 CPU가 아닌 다른 특정한 SW에 의해 번역되어서 기존 SW의 기능을 호출하거나 제어하는 명령들의 나열을 담고 있는 파일을 의미한다. RPA가 특정 업무를 자동화하려면 해당 정보시스템이 가진 여러 기능을 실행하는 일련의 명령어들을 조합해 내부적으로 스크립트 파일을 생성하고 실행한다.

(활용 효과) 직무(Job)가 아닌 작업(Task)을 자동화하기 때문에 SW를 이용해 업무능력을 증강할 수 있음

◦ (구현 시간 단축) RPA는 규칙적이고 정형적으로 반복되는 인간의 행동을 SW로 모방하는 방식이므로 SW개발보다 신속하게 자동화를 구현 가능

– 고비용의 비즈니스 요구사항 분석과 재작업이 필요하지 않고 기술 복잡도가 낮아 몇 주, 몇 달 만에 유효한 결과를 도출

◦ (저부가가치노동 감소) 반복적이고 노동집약적인 작업의 소요시간은 줄이되 정확도 및 품질은 올리며, 고부가가치 작업에 더 많은 시간과 핵심 역량을 할애할 수 있게 해 줌

– RPA는 주요 비즈니스 프로세스에 대한 처리량과 산출물의 품질을 높이는 저렴한 수단으로서 생산성을 증대하고 최종 결과물의 제공 가치를 증강함

– RPA 자동화는 직원이 중요한 비즈니스 활동 및 탐색 활동에 집중할 수 있도록 동기를 부여하고 업무만족도를 높임

◦ (업무프로세스 개선) RPA를 이용하면 결과의 정확도를 향상시키고 응답시간, 공정 주기를 단축할 뿐 아니라 측정 가능성과 투명성, 표준화 수준과 감사 가능성도 증대됨

[그림 11] RPA 적용 전 후 비교

출처 : https://www.ktds.com/competency/rpa_new.jsp, 2021.05.24. 접속

2. RPA 시장 지형과 전망

RPA 시장 규모는 ’21년 18.9억 달러에서 ’24년까지 연평균 두 자릿수로 성장하고, ’22년 글로벌 대기업 중 90%가 RPA를 도입할 전망(Gartner, ’20)

◦ (해외) 글로벌 기업들은 지능형 업무 자동화(IPA, Intelligent Process Automation)를 포함한 다양한 RPA 기능을 제공하며 시장을 선도하고 있음

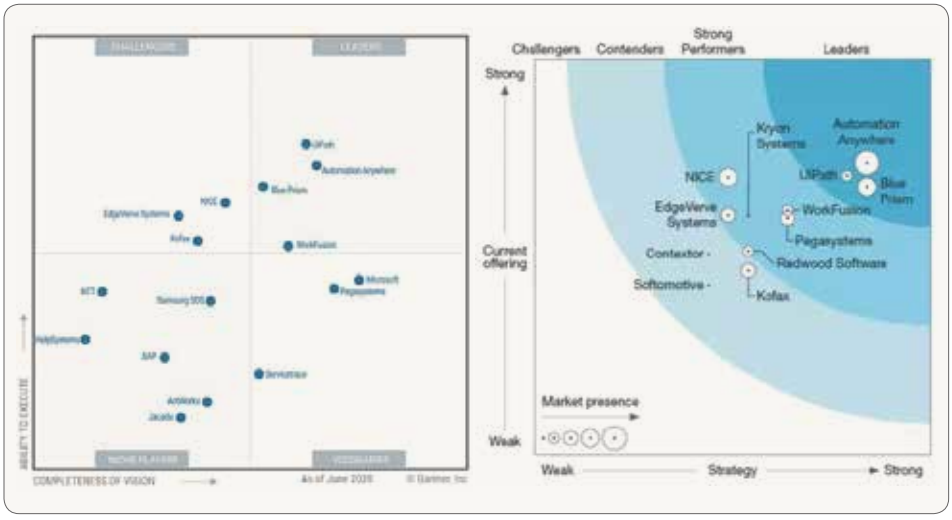
◦ 오토메이션애니웨어, 유아이패스, 블루프리즘, NICE, 페가시스템 등이 선도기업으로 ’18년 기준 글로벌 시장점유율 50%를 차지하고 있음

[표 7] RPA 제품/서비스 수준 비교(5점 척도)

워크퓨전	MS	오토메이션애니웨어	페가시스템즈	유아이패스	코팍스	서비스트레이스	NICE	엣지버브시스템즈	블루프리즘	SAP	삼성SDS
4.34	4.20	4.18	4.16	4.15	4.12	4.11	4.05	4.02	4.01	3.90	3.72

출처 : Gartner(’20)

[그림 12] 시장 참여자 지형도(좌: Gartner(’20), 우: Forrester(’17))



◦ (국내) ’21년 국내 RPA 시장은 1,000억 원 정도로 외산 솔루션과 국산 솔루션이 시장에서 경쟁하고 있음

– (외산 솔루션) 유아이패스, 오토메이션애니웨어 등이 진출한 상태임

– (국산 솔루션) 그리드원, 시메이션(체크메이트), 인지소프트, 이든티앤에스 등 국내 SW전문업체, 삼성SDS(브리티RPA), LG CNS, 포스코ICT(에이웍스) 등 IT서비스업체들이 제품을 출시하여 이미 활발하게 활동 중임

3. RPA 도입의 고려사항

도입기관이 목표하는 자동화·성숙도 수준에 따라 평가요소가 달라지는데, 일반적으로는 기술적 역량, 프로세스 관리, 운영 영향 및 위험관리 등으로 범주화됨

◦ (시범 도입 단계) 상대적으로 단순한 대상 업무 프로세스를 선정하고, 비용 요소, 사용 편의성, 기술 요소, 업체 지원을 종합 고려함

– (비용 요소) 시범 도입시의 초기 구축비용, 라이선스 비용, 운영비용 평가

– (사용 편의성) 코드/자동화 개발 편의성, 시스템 간 상호작용성·통합성 평가

– (기술 요소) OS/하드웨어 요구사항과 RPA 배치·운영에 필요한 기술적 역량

– (업체 지원) 지원, 교육 및 고객 서비스와 계약 사항 등을 평가

◦ (기술 아키텍쳐 수립) 명확한 장기 자동화 전략과 기술 요구 사항이 있을 때에 RPA 도입 평가를 추진하며 기능 검토 및 기본 사용 사례를 포함한 데모 수행

– (업체 경험) 업체의 시장 인지도, 동일 분야 기존 실적, 고객 사례, 계약 조건과 검토사항을 사전 평가함

– (제품 기능) 프로세스, 자동화, 운영 부문으로 나누어 평가함

[표 8] RPA 제품 기능의 상세 부문별 평가 요소

상세 기능	프로세스	자동화	운영
평가 요소	• 워크플로우 관리 • 프로세스 녹화 및 재현 • 자가 학습 역량 • 사용 편의성 • 프로세스 엔지니어링·평가	• 시각적 작성 도구 • 명령 라이브러리 • 완전/부분 자동화 역량 • 컴포넌트 공유 • 테스트/디버깅 통제 방식 • 사용 편의성	• 중앙집중형 배치, 관리, 스케줄링 • 라이센싱 구조 • 확장성, 가용성, 성능 관리 • 예외사항 관리 • 대시보드 역량 • 비즈니스, 운영 분석 기능

출처 : 美 Federal RPA Community of Practice(2020), RPA Program Playbook

– (보안) 애플리케이션 보안, 개인정보보호 등 법제도 컴플라이언스, 계정·개인 식별 관리, 위험/보안 평가, 인증, 데이터 암호화/보호, 프로세스 추적 가능성

– (아키텍쳐) HW/SW 요구사항 및 가상화 서버 설계, 멀티 테넌시, 온프레미스/클라우드, 권한, 가용성/재해복구 역량, 네트워크 용량/성능 관리 역량 검토

◦ (기술 정책) 보안, 계정·개인 식별 관리, 개인정보 보호 정책에 관한 검토 필요

– (보안 정책) RPA 구현의 범위를 명확히 정의하고 구현에 동반되는 위험을 평가한 후 위험 관리 전략과 권한 관리 전략, 코드 통제 전략 등을 수립

– (계정, 개인 식별 관리) 서비스/네트워크와 시스템/애플리케이션의 수준별 접근 관리가 필요

– (개인정보보호) 시스템/애플리케이션 명세, 역량·기능과 사용자를 검토하고, 상호작용하는 관련 시스템을 파악한 후 시스템이 취급하는 데이터 유형에 따라 현재 보안 정책 및 용례가 개인정보보호 정책을 위반하지 않는지 검토

◦ (프로세스 선택) 현업 인력이 직접 해오던 반복적이고 규칙화가 가능한 프로세스이면서, 많은 리소스가 소요되어 자동화 효과가 큰 프로세스를 식별하고 선택함

– 이미 수립된 기본 공통 자동화 사례나 타 기업·기관이 도입한 부문, 기존의 잘 알려진 비즈니스 이슈·도전 과제를 해결한 사례를 참조하여 우선순위를 식별함

* (예시 업무) 웹사이트, 앱의 로그인, 데이터의 입력, 웹의 숫자, 텍스트 등 데이터 추출, 이메일, 첨부파일 등 파일 열기, 옮기기, 정보 검색, 업데이트 등

[표 9] 프로세스 선택, 평가 및 개선을 위한 분석 단위 요소

평가 항목	주요 평가 요소	
적합성	복잡도	위치나 관련 조직의 수
		프로세스 정의/문서화 수준과 품질
		시스템과 애플리케이션 수
		연결/상호작용 유형(다수 시스템에의 접근과 연계성)
		스크린 수 및 입력 단계 수준(노동집약적/반복적)
		대량의 구조화된 데이터 입출력, 처리, 관리가 필요
		운영 준비도
전략부합성	기술	RPA 프로그램 용도 및 목표의 부합성
		RPA 프로그램 서비스 전달/운영 모델의 부합성
		RPA 프로그램 기술 정책/아키텍쳐의 부합성
전략부합성	전략	기업·기관의 미션과 목표
		리더십 우선순위와 전략, 이니셔티브
		부서단위 목표와 전사단위 목표에 부합하는 산출물

평가 항목	주요 평가 요소	
효과성	정량	노동시간 절감
		공정 주기 시간 단축
		처리량 증가 및 프로세스 산출물 증대 수준
	정성	컴플라이언스/감사 가능성의 개선
		전사 적용 가능성과 확장성
		정확도 향상

출처 : 美 Federal RPA Community of Practice(2020), RPA Program Playbook

◦ (운영 관리) 운영 관리, 변경 관리 및 자동화 중단 사고에 대한 대응, 라이센스 관리, 기술정책 업데이트, RPA 생명 주기 관리, 코드 공유 방식과 자동화 스케쥴링 등의 요소를 점검할 필요 있음

* 사람의 개입이나 의사결정이 필요한 업무 처리는 RPA 처리→사람의 처리→RPA 처리 등과 같은 약속된 업무 흐름을 정의할 필요 있음

4. RPA의 용도와 한계

(용도) RPA는 기업 및 기관 내 일반적인 반복업무에 적용 가능하지만, 내부 운영, 기능 개선, 위험관리·감사, 데이터 분석·보고 등 분야의 사례가 다수임

◦ (내부 운영) 재무회계, 인력관리, IT서비스, 구매, 행정·운영 효율화에 적용

◦ (기능 개선) 시스템 통합, 애드온을 통한 시스템 기능 개선, 데이터 검증 및 유효성 검사에 적용

◦ (위험관리·감사) 컴플라이언스, 작업증명·검토, 통제 자동화, 변경 관리, 위험 평가, 설문·조사 분석에 적용

◦ (데이터 분석·보고) 데이터 분석 자동 보고, 수집 및 클린징, 마이닝, 성능 모니터링 등에 적용

(한계) RPA는 특정한 업무에 한정한 개별 프로세스로 작동하고 도입효과가 큰 업무 분야와 업무를 선정하는데 많은 노력이 필요함

◦ RPA만을 사용해서 기업 및 기관의 모든 비즈니스 프로세스를 자동화하고 최적화하는 것은 어려우며, 적절한 프로세스 선택과 선진화·표준화에 더 많은 비용과 노력이 소요됨

V 요약과 시사

1. 본문의 요약

1980년대의 정보화혁명은 점점 심화되어 현재는 모든 기업 및 기관들이 SW 없이는 업무를 수행할 수 없는 디지털 전환의 시대가 도래함

◦ 특히 인터넷과 웹, 클라우드, SW를 자유롭게 설치하고 삭제할 수 있는 각종 모바일 기기의 등장으로 SW 개발 수요가 폭증하고 있음

– 1990년대에 인터넷과 웹은 웹프로그래밍 분야를 탄생시켰고 2000년대 중반 이후 SW를 자유롭게 설치/삭제할 수 있는 각종 모바일 기기가 윈도가 아닌 다른 운영체제들을 탑재하면서 운영체제가 다양화되었음

– 또한 기업 및 기관들의 정보시스템 환경이 클라우드로 바뀌면서 클라우드에 적합하게 기존 SW를 변경하거나 신규 개발할 필요도 증대되었음

◦ 이에 SW인력의 인건비가 상승하고 구인난이 심화되자 SW기업과 비SW기업들은 모두 보다 빨리 SW를 개발할 수 있는 다양한 방법들을 모색하고 있음

– 전 세계적으로 숙련된 SW개발자가 많이 부족한 상황이며 웹과 모바일 프로그래밍, 하이브리드 클라우드 환경 등에 효율적으로 대응하도록 기관의 IT 체계와 SW 구조를 새로 정립할 필요도 있음

이 리포트에서는 SW도구를 통해 SW개발의 생산성을 향상시키기 위한 그간의 대표적인 노력들을 정리하고 최근 화제가 되고 있는 로우코드·노코드와 RPA가 디지털 전환을 촉진할 수 있는 가능성과 한계에 관해 검토함

◦ 그래픽 사용자 환경에서 SW개발 생산성을 높이기 위한 RAD 도구가 등장 및 발전하였고, 프로그래밍 언어가 아닌 모델링 언어를 이용해 SW의 기능을 기술하고 소스코드를 자동 생성하려는 MDD 방식이 등장하여 현재도 계속 사용되고 있음

◦ 비즈니스 프로세스 구축·관리·최적화를 위한 BRE, BPM, BAM과 같은 도구, 데이터베이스 관리 및 최적화 보조 도구들이 등장하고 발전하여 로우코드·노코드와 RPA와 같은 도구들이 등장할 수 있는 기술적·사업적 배경이 되었음

전문 SW개발자가 아닌 숙련되지 않은 현업 직원들도 로우코드·노코드와 RPA를 활용하여 직접 문제를 해결할 수 있으나, 용도 등에서 일정부분 한계가 존재함

◦ (로우코드·노코드) 개발자가 프로그래밍 언어로 작성하는 부분을 최소화하거나 아예 필요 없도록 하여 현업 직원과 SW개발자가 손쉽게 SW개발을 할 수 있어 디지털 전환에 보다 신속히 대응하는 것을 목표로 함

- RAD 도구들이 SW개발자의 생산성을 올리는 것을 목표로 했다면, 로우코드·노코드는 현업직원의 SW개발을 활성화해 SW개발의 진입장벽을 낮추는 것을 목표로 한다는 점에서 차이가 있음
- 표현계층에 많은 노력과 예산이 소요되거나 규칙 기반으로 모델링 가능한 분야에 적합하나, 보수적인 운영 및 성능요건이 있는 환경에는 사용하기 어려우며 플랫폼 종속가능성이 높은 한계가 있음

◦ (RPA) 스크린 스크래핑 기술을 바탕으로 기존 SW들의 기능을 조합하여 단순 반복 작업을 자동화하는 스크립트를 생성하여 실행하며, 신규 SW를 개발한 것과 같은 효과를 볼 수 있음

- 단순반복 작업을 자동화해서 시간을 절약하고 정확도를 높이는 한편, 인력은 고부가가치노동에 집중할 수 있어 직무만족도가 올라갈 수 있음
- 특정 작업 별로 개별적으로 개선할 수 있어 전체 업무프로세스를 개선하기는 어려우며, 도입효과가 큰 업무분야를 선정하는데 노력이 필요함

디지털 전환 시대에 국내 기업과 수요기관들은 로우코드·노코드와 RPA에 보다 많은 관심을 기울일 필요가 있음

◦ 로우코드·노코드와 RPA는 잘 활용할 경우에는 낮은 비용으로도 필요한 SW를 개발하거나 작업들을 자동화하여 업무 생산성 향상에 크게 기여할 수 있으므로 수요기업과 기관들은 도입을 적극 검토하는 것이 필요함

◦ 국내 SW기업들은 로우코드·노코드와 RPA 모두 꾸준한 기술개발로 외산 솔루션 대비 기술 경쟁력을 키워야 하며, 플랫폼으로서 개발자 생태계 구축에도 많은 노력을 기울여야 함

2. 수요자인 기업과 정부/공공기관 관점의 시사점

(용도·한계) 로우코드·노코드와 RPA는 적절한 응용 분야와 용도가 있으며, 이를 통해 현업 직원들이 문제를 직접 해결하려면 컴퓨팅적 사고가 바탕이 돼야 함

◦ 로우코드·노코드는 실험적이거나 업무 생산성 향상을 위한 앱의 작성에 주로 사용되며, 미션 크리티컬하거나 보수적인 운영 및 성능 요건이 있는 환경에서는 사용되기 어렵고, 도입 기업이나 기관은 플랫폼에 종속될 가능성이 높음

◦ RPA는 특정 업무에 한정하여 적용되므로 기업 및 기관의 모든 비즈니스 프로세스를 자동화하고 최적화하는 것은 어려우며 적절한 프로세스 선택과 선진화·표준화에 더 많은 비용과 노력이 소요됨

◦ 이들은 숙련도가 낮은 현업 직원이 사용할 수 있는 도구이지만 기본적으로는 추상화, 자동화*를 통해 문제를 해결하는 컴퓨팅적 사고가 바탕이 되어야 함

* 컴퓨팅적 사고(computational thinking)란 컴퓨터가 처리할 수 있도록 문제와 해법을 표현하는 문제해결방법의 집합

* 추상화: 복잡하게 얽혀 있는 문제를 구조화하고 해결 가능한 상태로 만드는 것

* 자동화: 추상화된 문제를 컴퓨터의 언어로 바꾸는 것

(수요기업) 로우코드·노코드와 RPA의 적절한 조합 및 활용을 통하여 현업 직원들의 개발을 활성화하여 디지털 전환 역량을 증강할 수 있음

◦ (체계 정립) 기존의 IT부서와 적극적으로 소통하면서 로우코드·노코드와 RPA의 도입여부를 결정하고, 도입 시에는 전사 SW·ICT 아키텍처와 각 부서들의 역할을 재정립하여야 함

– 기존의 IT부서의 자체개발 업무에도 적용할지 등도 같이 결정해야 하며, 현업직원이 만드는 SW나 RPA 스크립트들의 점검 절차 및 방법도 확립해야 함

– 실제 수행하던 업무를 로우코드·노코드로 개발하거나 RPA를 활용해 자동화할 현업 인력들에게 제공할 인센티브도 검토 필요

◦ (인력관리) 업무효율화를 달성한다면 기존 인력이 보다 고부가가치의 직무를 수행할 수 있도록 재배치할 수 있음

– 현업 직원과 전문 SW개발자의 담당업무 및 필요역량을 재설계하고 각자 핵심 역량과 핵심 업무에 집중할 수 있도록 지원 가능

(정부·공공) 정부·공공도 로우코드·노코드로 가능한 SW 개발로 대체가능하거나 RPA 적용이 가능한 업무들을 검토하고 종사자 교육 등 활용을 시도해야 함

◦ 반복되는 단순 업무를 찾아내고 자동화하는 것은 디지털 전환의 기본이며, 정부·공공 내에도 자동화 가능한 단순 반복 업무가 존재할 가능성이 매우 높음

– 2019년에 한국남부발전이 RPA를 도입했으며, 2020년 행정안전부와 한국농수산식품유통공사는 국산 RPA 솔루션을 도입했고 2021년에는 한국전력공사도 동일 솔루션을 도입할 예정[6]으로 공공부문에도 RPA가 확산되고 있음

◦ 공무원 및 공공기관 직원들을 대상으로 기초적인 SW개발 역량과 RPA 사용법을 교육하면서 내부 업무의 효율화를 시도하는 것이 가능

◦ 로우코드·노코드 도구는 대부분 민간 클라우드 기반이나 온프레미스 용도 있고 RPA는 개인용도 존재하므로 시범사업 가능

3. 공급자인 SW기업 관점의 시사점

▨ 로우코드·노코드와 RPA의 전체 SW시장 내 비중은 1%로 적지만 성장가능성은 높고 국내 기업들의 진출도 가시화되고 있는 상황으로 꾸준한 기술개발이 필요함

◦ 로우코드·노코드 국산 솔루션은 태동기에 있어 시장의 니즈와 인식, 제품 인지도를 높이면서 장기적 관점에서 꾸준한 기술개발이 필요함

– 글로벌 강자의 솔루션을 벤치마킹하면서 제품의 완성도와 서비스의 경쟁력, 시장 인지도를 동시에 높여가는 것은 매우 도전적인 과제이므로 장기적 관점에서 지속적인 투자가 필요

◦ RPA의 경우 외산 솔루션과 국산 솔루션이 시장에서 경쟁하고 있는 상황으로 지속적인 제품 개선과 사용자 층 확대를 통해 내수 점유율을 높이고 글로벌 진출을 시도해 보는 것이 필요함

– 국내 기업들이 내수 점유율을 점점 늘려나가고 있는 상황이나, 국내 시장규모가 크지 않으므로 글로벌 기업들의 솔루션 대비 제품·서비스의 경쟁력을 높여서 해외 진출을 모색할 필요가 있음

◦ 고객 맞춤 기능을 제공하고 파트너 개입의 자유도를 높여 제품 성숙도를 높이는 한편 후발주자로써 틈새 시장을 노려볼 필요 있음

– 보편성을 추구하면서도 제품에 지역성(제도·문화·IT소비특성)과 같은 수요 특성을 반영하여 특정 응용 분야에서 필요한 템플릿, 사용사례 및 스토리 등을 개발 및 제공하면 국내외 시장에 진입할 가능성이 열려 있음

6 https://www.industrynews.co.kr/news/articleView.html?idxno=41942
그 외에도 SW개발역량을 갖춘 사회복무요원이 1년간 노동청에서 발송한 모든 등기우편을 조회해 종이에 인쇄하여 보관하라는 업무지시를 받았는데, 당초 오랜 시간이 소요될 것으로 예상되었지만 파이썬 라이브러리를 활용해 자동화하는 SW를 개발해 하루 만에 완료한 사례에 대해서는 다음 링크의 기사를 참조
https://www.donga.com/news/Society/article/all/20181218/93346855/1

– 금융·제조·공공 등 전통 산업 중 로우코드·노코드와 RPA에 친화적인 응용 분야를 발굴하고 모범사례를 확산할 필요 있음

– 적용 가능한 업무를 발굴하고 선택하는 설계 과정이 구현 과정보다 더 많은 비용·시간이 소모되므로 제품판매와 컨설팅을 병행할 필요 있음

장기적으로는 플랫폼이라는 점을 인식하여, 개발자와 사용자들 간 질의응답과 모범 사례를 공유하고 확산할 수 있는 커뮤니티와 교육훈련 체계를 구축하여 개발자·사용자와 함께 성장하는 사업모델을 구현해 내는 것이 중요함

◦ 이용자의 피드백을 제품 개선에 신속하게 반영하면서 자사 제품에 기반한 다양한 SW들을 탐색하고 구매 가능한 SW스토어를 구축할 필요 있음

◦ 교육훈련 체계를 수립하여 커뮤니티 네트워크와 연계함으로써 제품의 유통과 파트너의 관여·개입을 원활하게 하고 고객 맞춤 설정·기능 제공의 수단과 그러한 기능을 제공하는 주체의 범위를 파트너, 사용자까지 확대함

◦ 특히 현업 직원과 전문 SW개발자의 역할을 구분하고 서로 역할의 변화를 이해하고 원활하게 소통할 수 있도록 지원해야 함

1. 국내문헌

김기봉(2019), “업무 자동화를 위한 RPA 융합 기술 고찰”, 융합정보논문지, 9(7), 8-13.

진회승, 김태호(2015), “MDD(모델 주도 개발) 유용성 논의와 사례 분석”, SPRi 이슈리포트 2015-012호.

2. 국외문헌

Barr, D., Harrison, J., & Conery, L.(2011), “Computational thinking: A digital age skill for everyone”, Learning & Leading with Technology, 38(6), 20-23.

Core(2021), “2020 was a year of change: how ready was the market, and how ready were you?”.

Everest Group(2017), “How to Eliminate Enterprise Shadow IT”. https://www.everestgrp.com/2017-04-eliminate-enterprise-shadow-sherpas-blue-shirts-39459.html/.

Forrester(2020), https://go.forrester.com/blogs/category/low-code-platforms/.

Gartner(2017), “Make the best of Shadow IT”, https://www.gartner.com/smarter-withgartner/make-the-best-of-shadow-it/.

Gartner(2020), “Top 10 Trends in PaaS and Platform Innovation, 2020”.

IDG(2017), “Low Code”.

IEEE(2017), https://standards.ieee.org/standard/2755-2017.html.

ITIC(2017), “Hourly Downtime Tops $300K for 81% of Firms; 33% of Enterprises Say Downtime Costs >$1M”, https://itic-corp.com/blog/2017/05/hourly-downtime-tops-300k-for-81-of-firms-33-of-enterprises-say-downtime-costs-1m/.

KPMG(2020), “COVID-19 KPMG Business Report”.

Markets and Markets(2020), “Low-Code Development Platform Market by Component (Platform and Services), Application Type, Deployment Type (Cloud and On-Premises), Organization Size (SMEs and Large Enterprises), Industry, and Region - Global Forecast to 2025”.

Mckinsey(2020), “The State of AI in 2020”.

NATO(1969), P. Naur and B. Randell, (Eds.), “Software Engineering: Report of a conference” sponsored by the NATO Science Committee, Garmisch, Germany, 7-11 Oct. 1968, Brussels, Scientific Affairs Division.

Teece, D. J.(2007), “Explicating dynamic capabilities: the nature and microfoundations of (sustainable) enterprise performance”, Strategic management journal, 28(13), 1319-1350.

U.S. Bureau of Labor Statistics(2019), “Employment Projections 2019-2029”.

U.S. Federal RPA Community of Practice(2020), “RPA Program Playbook”.

참고문헌

Wing, J. M.(2008), “Computational thinking and thinking about computing”, Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences, 366(1881), 3717-3725.

3. 기 타

KT DS(2021), https://www.ktds.com/competency/rpa_new.jsp.

Mendix(2021), 홈페이지 비용 계산기. https://mendix.com.

Tobeware(2019), Robotic Process Automation.

동아일보(2018), https://www.donga.com/news/Society/article/all/20181218/93346855/1.

매일경제(2021), https://www.mk.co.kr/news/business/view/2021/03/242395/.

법률신문(2017), https://m.lawtimes.co.kr/Content/Article?serial=119107.

에너지데일리(2019), http://www.energydaily.co.kr/news/articleView.html?idxno=103788.

인더스트리뉴스(2021), https://www.industrynews.co.kr/news/articleView.html?idxno=41942.

전자정부 표준 프레임워크 강의.

중앙선데이(2021), https://news.joins.com/article/24026904.

지능정보화 기본법[시행 2020. 12. 10.] [법률 제17344호, 2020. 6. 9., 전부개정].