DOI: 10.15514/ISPRAS-2021-33(3)-8

Power Fx: Low-code Language for Collaboration Tools

I.A. Voronkov, ORCID: 0000-0001-5620-1002<iliaftk@outlook.com>
S.E. Saradgishvili, ORCID: 0000-0002-1291-1675 <ssarad@ya.ru>

Peter the Great St.Petersburg Polytechnic University
29, Polytechnicheskaya, St.Petersburg, 195251, Russia

Abstract. The paper provides an overview of the first impression of the language for implementation low code of approach. About a month has passed since the release date.

Keywords: low-code; collaboration; power platform

For citation: Voronkov I.A., Saradgishvili S.E. Power Fx: Low-code Language for Lollaboration Tools. Trudy ISP RAN/Proc. ISP RAS, vol. 33, issue 3, 2021, pp. 101-108. DOI: 10.15514/ISPRAS-2021-33(3)-8

Power Fx: Low-code язык для инструментов совместной работы

И.А. Воронков, ORCID: 0000-0001-5620-1002<iliaftk@outlook.com>
С.Э. Сараджишвили, ORCID: 0000-0002-1291-1675 <ssarad@ya.ru>

Санкт-Петербургский Политехнический университет Петра Великого,
Россия, Санкт-Петербург, 195251, ул. Политехническая, д. 29

Аннотация. В статье представлен обзор первого впечатления от языка для реализации подхода low code. Со дня релиза прошло около одного месяца.

Ключевые слова: автоматизированное тестирование; обеспечение качества; статический анализ исходного кода.

Для цитирования: Воронков И.А., Сараджишвили С.Э. Power Fx: Low-code язык для инструментов совместной работы. Труды ИСП РАН, том 33, вып. 3, 2021 г., стр. 101-108 (на английском языке). DOI: 10.15514/ISPRAS–2021–33(3)–8.

1. Introduction

The IT market is now experiencing a new round of its rapid development. The demand for specialists in this area remains at least high [1].

According to the rules of the market, the cost of projects and services is growing following demand. Numerous platforms operating on the principles of low-code and zero-code have become one of the options for solving problems for automation and digitalization [2]. The general approach in such solutions is that most of the processes can be represented in the form of a graphical designer that works in accordance with UML/BPMN. The main feature is that the developer does not need to think about how interactions occur at the level of data structures, what algorithms are used for selection, sorting, and how computations are parallelized. Main objective: Implementation of the business requirement by creating a process state diagram. From a business point of view, this

paradigm is the most effective way to solve a problem, since business users are accustomed to thinking precisely by the tasks that a particular system must solve.

Unfortunately, at this stage of technology development, we are not able to completely switch to such technological solutions. The reasons may be different: the inability to completely solve the problem using only design tools, the lack of specialists with modeling, programming and business experience, the presence of a large layer of legacy code. Nevertheless, it is worth noting the growth of such decisions not only in business, but also in the scientific community [3, 4]. Along with the development of technologies, the question of the place of these solutions in the future is becoming more acute.

Many owners of large IT companies are promoting the idea that programming and software development should cease to be a highly specialized area [5]. The main idea is that in the future, people will use software tools to solve their personalized tasks. To do this, you do not have to get an education, take special courses. All you need to know is what you want to achieve with the program and how it can make your life easier. Of course, such ideas may seem utopian and unrealizable on the horizon of the next 10–20 years but looking back at the history of the development of the IT industry, we begin to think that all this may happen in the near future [6].

In this article, we provide an overview of an intermediate: a language that can empower people who are familiar with Excel syntax to develop software.

2. Description

At the beginning, we emphasize that this language is part of the Power platform. This is a relatively new vision of Microsoft corporation about the business data warehouse as a single point of connection and storage of data and tools. Together with the Power platform, the Power automate product demonstrates its development - a workflow designer, partly the successor to SSIS, SharePoint workflow engine. Together, these tools are able to close most of the tasks of automating enterprise activities.

To enhance the ability to develop using third-party technologies and programming languages, Microsoft introduced the Microsoft Graph API [7]. Microsoft Graph is the gateway to data and intelligence in Microsoft 365. It provides a unified programmability model that you can use to access the tremendous amount of data in Microsoft 365, Windows 10, and Enterprise Mobility + Security. Use the wealth of data in Microsoft Graph to build apps for organizations and consumers that interact with millions of users. The Microsoft Graph API offers a single endpoint, https://graph.microsoft.com, to provide access to rich, people-centric data and insights in the Microsoft cloud, including Microsoft 365, Windows 10, and Enterprise Mobility + Security. You can use REST APIs or SDKs to access the endpoint and build apps that support Microsoft 365 scenarios, spanning across productivity, collaboration, education, people and workplace intelligence, and much more.

Microsoft Graph also includes a powerful set of services that manage user and device identity, access, compliance, security, and help protect organizations from data leakage or loss. Microsoft Graph connectors (preview) work in the incoming direction, delivering data external to the Microsoft cloud into Microsoft Graph services and applications, to enhance Microsoft 365 experiences such as Microsoft Search. Connectors exist for many commonly used data sources such as Box, Google Drive, Jira, and Salesforce. Microsoft Graph data connect provides a set of tools to streamline secure and scalable delivery of Microsoft Graph data to popular Azure data stores. The cached data serves as data sources for Azure development tools that you can use to build intelligent applications.

Three key traits of Power Fx:

- The future of programming is open. Microsoft has embraced the pace of open innovation that has accelerated the adoption of languages like C# and Typescript. With Power Fx. Microsoft will open-source Power Fx, making the language available for open contribution by the broader community on GitHub.

- Power Fx is based on Microsoft Excel. Using formulas that are already familiar to hundreds of millions of users, Power Fx allows a broad range of people to bring skills they already know to low code solutions. Power Fx becomes a common ground for business users and professional developers alike to express logic and solve problems.
- Power Fx is built for low code. Power Fx is already the foundation of the Microsoft Power Apps canvas.

2.1 Data types [8]:

- **Boolean**. A true or false value. Can be used directly in If, Filter and other functions.Example: *false*
- **Color**. A color specification, including an alpha channel. Example: *ColorValue("#102031")*
- **Currency**. A currency value that's stored in a floating-point number. Example: *333*
- **Date**. A date without a time, in the time zone of the app's user. Example: *Date(2021, 5, 16)*
- **DateTime**. A date with a time, in the time zone of the app's user. Example: *DateTimeValue("May 21, 2019 11:00:09 PM")*
- **GUID**. A Globally Unique Identifier. Example: *GUID()*
- **Hyperlink**. A text string that holds a hyperlink. Example: m*ake.powerapps.com*
- **Image**. A Universal Resource Identifier (URI) text string to an image in .jpeg, .png, .svg, .gif, or other common web-image format.
- **Media**. A URI text string to a video or audio recording.
- **Number**. A floating-point number. Example: *8.903e121*
- **Option set**. A choice from a set of options, backed by a number. This data type combines a localizable text label with a numeric value. The label appears in the app, and the numeric value is stored and used for comparisons. Example: *ThisItem.OrderStatus*
- **Record**. A record of data values. This compound data type contains instances of other data types that are listed in this topic. More information: Working with tables.
- **Record reference**. A reference to a record in an entity. Such references are often used with polymorphic lookups.
- **Table**. A table of records. All of the records must have the same names for their fields with the same data types, and omitted fields are treated as blank. This compound data type contains instances of other data types that are listed in this topic. More information: Working with tables.
- **Text**. A Unicode text string. Example: *"Hello, World"*
- **Time**. A time without a date, in the time zone of the app's user. Example: Example: *Time(12, 13, 35)*
- **Two option.** A choice from a set of two options, backed by a boolean value. This data type combines a localizable text label with a boolean value. The label appears in the app, and the boolean value is stored and used for comparisons.

This set of types allows you to store business information of any complexity. Expansion of primitives is planned to support translation of types from one to another.

2.2 Working with datasets

Power Fx supports the If / else construct, but first-time users of this tool may be confused by the lack of a For loop. We tend to attribute this semantic idea to the fast-paced idea of promoting LINQ expressions [9]. Working with sets is based on the following commands:

- **Clear**. The Clear function deletes all the records of a collection. The columns of the collection will remain.Note that Clear only operates on collections and not other data sources. You can use RemoveIf(DataSource, true) for this purpose. Use caution as this will remove all records from the data source's storage and can affect other users. You can use the Remove function to

selectively remove records.Clear has no return value. It can only be used in a behavior formula. Example: Clear(DataSource).

- **ClearCollect**. The ClearCollect function deletes all the records from a collection. And then adds a different set of records to the same collection. With a single function, ClearCollect offers the combination of Clear and then Collect.
- **Delegation**. When used with a data source, these functions can't be delegated. Only the first portion of the data source will be retrieved and then the function applied. The result may not represent the complete story. A warning may appear at authoring time to remind you of this limitation and to suggest switching to delegable alternatives where possible. Nevertheless, if it is necessary to organize a For loop to iterate over the data, we can use the extension mechanism. . The most common use case for this mechanism can be represented in the form of interaction with the Microsoft Teams component, where the message acts as a triggering event. Fig. 1 demonstrates a step for triggering an event with receiving/ sending a message to a special channel.

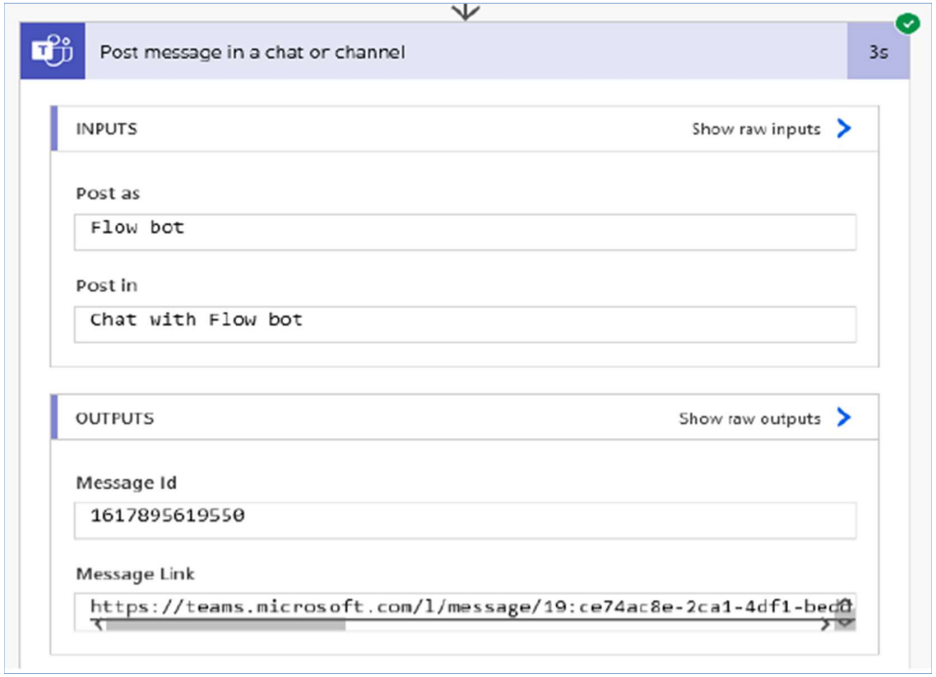

Fig. 1. Send teams message

In this example (listing 1), we can observe the possibility of traversing the collection with a search by values from another source. The key feature here is the fact that the data types in collections can differ in both type and number of values. One of the most common examples is searching on columns with multiple values (User () AAD) [10].

```
CountRows(
            Filter(
                ForAll(
                    'TestArea.Value,
                    If(
                        Value in TArea1.SelectedItems.Value,
                        {t: 1},
                        {t: 0}
                    )
```

```
                ),
                t = 1
            )
        ) >= CountRows(TArea1.SelectedItems.Value),
        CountRows(
            Filter(
                ForAll(
                    Product.Value,
                    If(
                        Value in ProductFilterChoice.SelectedItems.Value,
                        {x: 1},
                        {x: 0}
                    )
                ),
                x = 1
            )
        ) >= CountRows(ProductFilterChoice.SelectedItems.Value)
If(
    !IsBlank(CommentInput),
    Patch(
        TEST_Comments,
        Defaults(TEST_Comments),
        {
            TEST_Comments_ID_BP: displayItem.ID,
            TEST_Comments_Comment: CommentInput.Text,
            Title: CommentInput.Text,
            TEST_Comments_User: {
                '@odata.type':
"#Microsoft.Azure.Connectors.SharePoint.SPListExpandedUser",
                Claims: "i:0#.f|membership|" & Lower(User().Email),
                Department: "",
                DisplayName: User().FullName,
                Email: User().Email,
                JobTitle: ".",
                Picture: "."
            }
        }
    );
    Collect(
        Comments,
        {
            Title: CommentInput.Text,
            Created: Now(),
            TEST_Comments_User: {
                '@odata.type':
"#Microsoft.Azure.Connectors.SharePoint.SPListExpandedUser",
                Claims: "i:0#.f|membership|" & Lower(User().Email),
                Department: "",
                DisplayName: User().FullName,
                Email: User().Email,
                JobTitle: ".",
                Picture: "."
            }
        }
    );UpdateIf(
    TestSPCollection,
    ID = displayItem.ID,
    {
```



```
        Number_of_Comments: CountRows(Comments)

    }
);
    Reset(CommentInput);

)
collection_originalStatuses = ["New", "Pending", "Complete"]


ForAll(
// loop through the original collection
collection_originalStatuses,
// copy each one to a new collection
Collect(
collection_indexedStatuses,
// create an object with the value (or other props) and an index
{
Value: Value,
// the index starts at 0 and increments as each item is copied
Index: CountRows(collection_indexedStatuses)
}
)
)
collection_indexedStatuses = [
{Value: "New", Index: 0},
{Value: "Pending", Index: 1},
{Value: "Complete", Index: 2}
]
```

Listing 1. Traversing a collection

3. Interim results of the research

In this paper, we have described the main features of the language and ways to expand the functionality. To automate tasks in enterprises, this language is well used, and there is also a practice of reusing components: connectors, XML code. To compare the speed of delivery development, we plan to implement an experiment on the implementation of typical tasks using this language. On a specific example of updating the structure of a multiple field with users and their profiles, the language, as well as the low-code approach, showed excellent results.

References

[1] Rezaee Jordehi. Dynamic environmental-economic load dispatch in grid-connected microgrids with demand response programs considering the uncertainties of demand, renewable generation and market price. International Journal of Numerical Modelling: Electronic Networks, Devices and Fields, vol. 34, no. 1, 2021, 17 p.

[2] R. Sanchis, Ó. García-Perales, F. Fraile, R. Poler. Low-Code as Enabler of Digital Transformation in Manufacturing Industry. Applied Science, vol. 10, no. 1, 2019, 12 p.

[3] M.-L. How. Artificial Intelligence for Social Good in Responsible Global Citizenship Education: An Inclusive Democratized Low-Code Approach. In Proc. of the 3rd World Conference on Teaching and Education, 2021, pp. 81-89.

[4] N. Rauschmayr et al. Amazon SageMaker debugger: a system for real-time insights into machine learning model training. In Proc. of the 4th MLSys Conference, 2021, 13 p.

[5] E. Straschnov. You Shouldn't Have to Learn How to Code. Available at https://www.huffpost.com/entry/you-shouldnt-have-to-lear_b_6111914, accessed Apr. 08, 2021.

[6] L. Floridi and M. Chiriatti. GPT-3: Its Nature, Scope, Limits, and Consequences. Minds and Machines, vol. 30, no. 4, 2020, pp. 681-694.

[7] Overview of Microsoft Graph. Available at https://docs.microsoft.com/en-us/graph/overview, accessed Apr. 08, 2021.
[8] Microsoft Power Fx. Data types. Available at https://github.com/microsoft/Power-Fx/blob/main/docs/data-types.md, accessed Apr. 08, 2021.
[9] Y. Bai. Introduction to Language Integrated Query (LINQ). In SQL Server Database Programming with Visual Basic.NET, Wiley, 2020, pp. 123-213.
[10] T. Shimayoshi, Y. Kasahara, and N. Fujimura. Challenge for Consolidation of Individual Email Services into a Cloud Service. In Proc. of the ACM SIGUCCS Annual Conference, Mar. 2021, pp. 26-29.

Информация об авторах / Information about authors

Илья Александрович ВОРОНКОВ, аспирант. Научные интересы: платформы для совместной работы, ETL.

Ilia Alexandrovich VORONKOV, PhD Student. Research Interests: collaboration platforms, ETL.

Сергей Эрикович САРАДЖИШВИЛИ, кандидат технических наук, доцент. Научные интересы: обработка многомерных сигналов, комплексные системы.

Sergey Erikovich SARADGISHVILI, Candidate of Technical Sciences, Associate Professor. Research interests: multidimensional signal processing, complex systems.