EVALUATION OF FLEXIBLE POINTS ON USER INTERFACE FOR INFORMATION SYSTEM

Limin Shen, Chunyan Gao and Wenwen Jiang
Department of Computer Science and Technology, Yanshan University, Qinhuangdao, China

Keywords: User interface, flexible point, flexible change, flexible degree, function point.

Abstract: To deal with user requirement changes at runtime, information system software provides adaptable operations through user interfaces to change software functionality. We suggest the FleXible Point(FXP), flexible changes, flexible degree, flexible force, and flexible distance, to evaluate the effect of such user interfaces. Flexible degree is determined by flexible distance and flexible force. Flexible distance is measured by function point counting and flexible force is measured by level value of flexible point. An application to wage calculation software is given to illustrate the evaluation and measurement based on the FXP. The approach can be used as a guide to adjusting, improving, and to comparing the FXP on user interfaces, and arranging different levels of manipulators to increase the FXP efficiency and to bring the FXPs on user interface into play.

1 INTRODUCTION

To deal with user requirement changes, some software products provide adaptable operations through flexible or intelligent user interface (UI). We define such a point or a location on UI as a FleXible Point (FXP), and define the changes triggered and caused by the FXP as flexible changes.

What is flexible? Thesaurus Dictionary gives the following interpretations: "Capable of being bent, turned, bowed, flexed repeatedly without breaking, injury or damage; capable of being adapted; responsive to change; adaptable". Accordingly, the flexible changes in software are controllable, repeatable, inversed, consistent and harmonious.

The FXPs and flexible changes have made software system more flexible and software can change as user requirements change. How to evaluate the effect of FXPs on UI and the flexibility brought by them is to yet be clearly defined. We present an approach based on the FXP and flexible change to evaluate the effect of the FXP and its flexibility brought by it.

2 RELATED WORK

In software engineering, software flexibility concept had appeared in 1979. Pamas (1979) thought flexible software is one that can be easily changed, extended, contracted, or else in order to be used in a variety of ways. Keith Bennett (1999, 2000) presented a list of features about flexible software, suggesting that flexible software should be necessary and sufficient, personalized, adaptable and self adaptive, distributed, in small units and transparent. Amnon Eden (2006) presented evaluating software flexibility from program paradigm, architecture and design patterns. Robin Jeffries (1991) suggested that a UI was evaluated prior to its release by heuristic evaluation, software guidelines, cognitive walkthroughs, and usability testing.

However, most research in this area concentrates on evaluation of UI design, usability and reliability. Few research aims at providing a model that allows discussing and expressing quantitative relations between UI and software flexibility.

3 MEASUREMENT FACTORS

3.1 Changes via Force

To study how software can be changed, the "force" concept is introduced. It is the force that pushes software change. The force F consists of external force F_E and internal force F_I, $F = F_E + F_I$. The internal force F_I is given by software itself at

Shen L., Gao C. and Jiang W. (2007).
EVALUATION OF FLEXIBLE POINTS ON USER INTERFACE FOR INFORMATION SYSTEM.
In *Proceedings of the Ninth International Conference on Enterprise Information Systems - HCI*, pages 253-256
DOI: 10.5220/0002347002530256

runtime; the external force F_E is given by a manipulator through a FXP. F is determined by software internal structure, such as architecture style, design pattern, framework, etc. F_I is determined by flexible and adaptive mechanism such as dynamic binding, reflection and control platform, etc.

3.2 Definition of Concepts

The quantitative concepts are introduced as follows:

. **Flexible Point *FXP$_i$*:** a point or a location on the interface that can cause and trigger flexible changes to occur, through which F_E may apply.

. **Flexible Force** f_i **:** the minimum F_E applied to FXP_i that may trigger software to change.

. **Flexible Distance** S_i**:** the maximum range or size of software change caused by f_i through *FXPi.*

. **Flexible Degree** K_i **:** $K_i = S_i/(1+f_i)$, a measure for software flexibility brought by FXP_i.

. **Flexible Capacity** C : $C = \sum_{i=1}^{N} K_i$, a measure of entire or partial flexibility brought by a set of FXPs.

4 ANALYSIS BASED ON FXPS

Software manipulators are divided into three levels: Low-level User (LU), High-level User (HU) and Developer-level User (DU). Their F_E is F_{LU}, F_{HU} and F_{DU} respectively, obviously, $F_{LU} \le F_{HU} \le F_{DU}$. Whether a manipulator can utilize a FXP_i is determined by the fact whether $F_E > f_i$. The FXPs can be distinguished into four levels based on the relation between f_i and F_E.

Self-Adaptive FXP (SAFXP): all FXP_i with $f_i = 0$, oriented to all users. The flexible capacity brought by the SAFXP is

$$C_{SAFXP} = \sum_{i=1}^{N} K_i \mid f_i = 0 \quad (1)$$

LU-oriented FXP (LUFXP): all FXP_i with $0 < f_i \le F_{LU}$, oriented to the LU.

$$C_{LUFXP} = \sum_{i=1}^{N} K_i \quad \mid 0 < f_i \le F_{LU} \quad (2)$$

HU-oriented FXP (HUFXP): all FXP_i with $F_{LU} < f_i \le F_{HU}$, oriented to the HU.

$$C_{HUFXP} = \sum_{i=1}^{N} K_i \quad \mid F_{LU} < f_i \le F_{HU} \quad (3)$$

DU-oriented FXP (DUFXP): all FXP_i with $F_{HU} < f_i \le F_{DU}$, oriented to the DU.

$$C_{DUFXP} = \sum_{i=1}^{N} K_i \quad \mid F_{HU} < f_i \le F_{DU} \quad (4)$$

Related factors of the FXP have relations:
Implementation Difficulty (*ID*):
$ID_{SAFXP} > ID_{LUFXP} > ID_{HUFXP} > ID_{DUFXXP;}$
. Manipulation Easiness (*ME*):
$ME_{SAFXP} > ME_{LUFXP} > ME_{HUFXP} > ME_{DUFXP;}$
. Manipulator Cost (*MC*):
$MC_{SAFXP} < MC_{LUFXP} < MC_{HUFXP} < MC_{DUFXP}$.

5 EVALUATION OF FXPS

The necessary measure steps of FXPs are as follows:

(1) Identify the FXP. (2) Calculate the FXP's flexible distance. (3) Determine flexible force value of the FXP. (4) Calculate the flexible degree of the FXP. (5) Calculate flexible capacity for different analysis.

5.1 Identification of FXPs

Our initial investigation classified the FXP into five categories. (1) The adjustment of input interface: software users can customize or modify the input manners, contents and formats, etc. (2) The adjustment of output interface: software users can customize or modify the output manners, contents and formats. (3) The adjustment of business rules: the users can maintain the business rules through FXPs. (4) The adjustment of business flows: according to the actual requirements users can adjust the business flows. (5) The adjustment of data structure and data source: users can select data sources, and modify data structures, data ranges, etc.

The FXPs can be menu items, command buttons, textboxes, drop-down-lists, tables, graphs, etc. For general measurements, they only need provide exercisable software and user instructions.

There are some FXPs instances often used on UI: (1) adjust the value range of data element; (2) add /delete business rules; (3) add/delete items in selection; (4) add calculation formulas; (5) add/delete information items; (6) adjust screen

layout; (7) change data items type; (8) Select data sources; (9) find services automatically or adaptively.

5.2 Calculation of Flexible Distance

We use function point count as unit of measuring flexible distance S_i. Function point counts can represent the functional size of software in the users' view, which have four main advantages as follows. Firstly, it is easier to locate, identify and determine the changes, for function point method classifies software function into five components: EI (external input), EO (external output), EQ (external inquiry), ILF (internal logic files) and EIF (external interface files). Secondly, we can directly use the rules given by IFPUG. Thirdly, the function point analysis (FPA) provides measurement rules for GUI. Finally, the FPA is independent of platform and program languages.

We adjusted the process of FPA for calculation flexible distance as follows: (1) determine the application boundary of the FXP; (2) identify and rate transactional function types to determine their contribution to the Unadjusted Function Point (UFP) count; (3) identify and rate data function types to determine their contribution to the UFP count; (4) take UFP counts as flexible distance.

Table 1: Flexible force value of FXP.

FXP Level	f_i	Manipulation
SAFXP	0	No user's manipulation
LUFXP	10	Simple functional manipulation
HUFXP	20	Functional and business manipulation
DUFXP	30	Average technical manipulation

5.3 Determination Flexible Force Value

The value of flexible force f_i is determined by the FXP level. We defined the flexible force value of self-adaptive flexible point is 0. The scale value of each level is defined as 10. Because of f_{SAFP}=0 and $f_{SAFXP} < f_{LUFXP} < f_{HUFXP} < f_{DUFXP}$, we can define the flexible force as table 1.

5.4 Calculation Flexible Degree and Capacity

Once flexible force f_i and flexible distance S_i are gained, the flexible degree of FXP_i can be calculated by the formula $K_i = S_i /(1+f_i)$. After each FXP's flexible degree is determined, it is time to calculate different types of flexible capacity.

6 CASE STUDY

A Wage Calculation Software (WCS) has the elementary functions: data input, data print, inquiry, data import and data export. Meanwhile, the WCS provides six FXPs, which are shown as the second column on Table 3.

We proposed three implementation schemes. The level of FXP in each scheme may be different, which is indicated at column f_i. On Table 3, if $f_i = \infty$, it means the FXP does not exist.

In scheme 1, FXP_2 is a HUFXP, which is suitable for HU to add or delete information items such as address, e-mail, birth date, and calculable items such as traffic allowance, worked hours without modifying codes. At this point, the software gives a relatively simple manipulation screen to the end user. After users add or delete wage items, the WCS is able to automatically adjust and change data input interface, data output interface, and internal logical files. Function point components EI, EO, EQ, ILF and EIF are impacted and changed. While in schemes 2, FXP_2 is designed as a DUFXP, it needs developer's intervention to satisfy requirements above, and the flexibility at the FXP decreases, but its implementation mechanism will be simple. In schemes 3, FXP_2 does not exist, when the requirements above change, users and the maintainers have to change software codes.

We assume that prepared manipulators for the WCS are LU and HU. Available FXPs (AFXP) are FXPs which users have ability to manipulate.

The capacity of the AFXP in the case is

$$C_{AFXP} = \sum_{i=1}^{N} K_i \mid f_i \le F_E = \sum_{i=1}^{N} K_i \mid f_i \le F_{HU} = C_{LUFXP} + C_{\mathrm{HUFXPP}} \quad (5)$$

Rate of FXP's availability (RA) is

RA = number of the AFXP/ total number of the FXP (6)

Table 2 shows flexible capacity of every scheme. Scheme 1 gains the highest flexibility and the highest RA because prepared manipulators accord with required manipulators. Inversely, scheme 3 gains the lowest flexibility and RA is 0. The data on Table 3 and Table 2 can be used as quantitative information to guide software developers and users to improve the UI intelligence and software flexibility.

Table 2: The calculation of flexible distance, flexible force and flexible capacity.

No	Flexible points FXP_i	Changed function components	S_i	Scheme 1		Scheme2		Scheme 3	
				f_i	K_i	f_i	K_i	f_i	K_i
1	Adjust the wage item width	EI, EO, EQ, ILF,EIF	56	20(HUFXP)	2.67	20(HUFXP)	2.67	30(DUFXP)	1.81
2	Add/delete wage item	EI,EO,EQ,ILF,EIF	178	20(HUFXP)	8.48	30(DUFXP)	5.74	∞	0
3	Modify calculation formulas	EO, ILF	29	10(LUFXP)	2.64	20(HUFXP)	1.38	30(DUFXP)	0.94
4	Adjust screen layout	EI	6	0(SAFXP)	6	20(HUFXP)	0.29	30(DUFXP)	0.19
5	Add calculation formulas	EO, ILF	30	20(HUFXP)	1.42	30(DUFXP)	0.97	∞	0
6	Adjust wage print table	EO, EQ	12	10(LUFXP)	1.09	20(HUFXP)	0.57	30(DUFXP)	0.39

In the measurement process, we found that though FXPs in scheme 3 were difficult to manipulate, they were visible to the end user. Some of the FXPs in scheme 1 are easy to manipulate but they are hidden deeper and it is difficult to find them. Thus the visibility of the FXP and its manipulation difficulty have crucial influence on the usability of the FXP. So, the metric should have considered this issue.

Table 3: Analysis of WCS' FXP.

Flexible capacity	Scheme 1	Scheme2	Scheme 3
SA flexible capacity	6	0	0
LU flexible capacity	3.73	0	0
HU flexible capacity	12.57	4.91	0
DU flexble capacity	0	6.71	3.33
Available flexible capacity	22.3	4.90	0.00
Available rate of FXP	100%	42.25%	0.00%
* Manipulators are the LU and the HU			

7 CONCLUSION

This is our initial investigation on evaluation of the FXPs on UI. We put forward a new concept FXP, built a measurement model with it, and summarized how to quantify the FXPs. Though the measurement can't solve all the fundamental problems on software flexibility, our work provides a new way to understand and answer questions about software flexibility.

REFERENCES

K. Bennett, P. Bennett and D. Budgen, 2000. Service Based Software: The Future for Flexible Software, Proceeding of Asia-Pacific Software Engineering Conference (APSEC'00), Singapore.

Thesaurus Dictionary, http://thesaurus.reference.com/

D. Pamas, 1979. Designing software for Ease of Extension and Contraction. IEEE Transactions on Software Engineering. Vol SE-5, No 2

P. Brereton, D. Budgen and K. Bennett, 1999. The Future of Software, Communication of The ACM. Vol 42, No12

Robin Jeffries, 1991.User interface evaluation in the real world: a comparison of four techniques**,** Proceedings of the SIGCHI Conference on Human Factors in Computing Systems: Reaching through Technology.

A.H. Eden and T. Mens, 2006. Measuring software flexibility .IEE Proceedings Software. Vol 153, No 3

Function Point Training Manual, www.SoftwareMetrics.Com